Michael Halvorson

Opeta itsellesi Microsoft Visual Basic 5

***Suomen Atk**-kustannus*

Opeta itsellesi Visual Basic 5

Copyright © 1997 byMicrosoft Corporation

Original English language edition Copyright © 1997Michael Halvorson

All rights published by arrangement with the original publisher, Microsoft Press, a division of Microsoft Corporation, Redmond, Washington, USA

Kaikki oikeudet pidätetään. Tämän julkaisun jäljentäminen ilman tekijän kirjallista lupaa painamalla,monistamalla, äänittämällä tai muulla tavoin on tekijänoikeuslainmukaisesti kielletty.

Laikki mahdollinen on tehty kirjan tietojen oikeellisuuden varmistamiseksi. Microsoft Press ja Suomen Atk-kustannus Oy eivät vastaa kirjan tarkkuudesta ja kattavuudesta sekä niistä aiheutuneista seurauksista.

Suomenkielisen version on julkaissut SuomenAtk-kustannus Oy Copyright © 1997.

Alkuperäinen nimi:MicrosoftViual Basic 5 Step by Step

ISBN 951-762-528-6

Kirjoittaja	Michael Halvorson
Kääntäjä	Veli-Pekka Ketola
Kansi	Frank Chaumont Timo Ravela (sähköinen versio)
Kustantaja	Suomen Atk-kustannus Oy PL 68 02601 ESPOO Puhelin (09) 4765 8550 Telekopio (09) 4765 8595 Sähköposti: palvelu@satku.fi Internet: www.satku.fi
Sähköinen laitos	Suomen Atk-kustannus Oy 1997

Kiitokset

Kirjoitin suurimman osan tästä kirjasta viihtyisässä toimistossa, joka sijaitsee lähellä Queen Anne Hillin huippua Washingtonissa. Olen erityisesti kiitollinen vaimolleni Kimille, joka tuki minua voimallisesti jälleen yhden kirjan kypsyttelyvaiheen aikana ja joka yllätti minut aika ajoin vierailemalla luonani poikamme Henry Jamesin kanssa. Tätä kirjoittaessani Henry kolistelee ikivanhaa näppäimistöä, joka lojuu irrallaan toimistoni lattialla: kop, kop, naks.

Joukko Microsoft Pressin lahjakkaita ammattilaisia ohjasi tämän kirjan sekä suvantojen että virtapaikkojen ohi. Casey Doyle aloitti muokkaustyön ja esitteli minut Visual Basic -ryhmän asialleen vihkiytyneille ohjelmoijille, testaajille ja toimittajille. He esittivät kommenttinsa jokaisen harjoituksen ohjelmista ja ohjeista. Lisa Theobald sopeutui kärsivällisesti monta kirjaa sisältävään kirjoitusaikatauluuni ja opasti käsikirjoituksen toimitus- ja tuotantoprosessin läpi. Jean Ross analysoi ohjelmalistaukset ja tekniset kuvaukset huolellisesti ja korjasi kömpelöimmät virheeni. Bill Teel teki tavanomaisia taikatemppujaan elektronisten piirrosten muodossa. Pam Hidaka hienosääti kirjan alkuperäisen ulkoasun, jonka oli laatinut Kim Eggleston. Peggy Herman, Elisabeth Thébaud Pong ja Linda Robinson taittoivat kirjan kauniisti. Lopuksi Microsoft Pressin oikaisulukijat etsivät orporivit, kirjoitusvirheet, väärin sijoitetut ajatusviivat ja puuttuvat sivut aivan verrattomalla tavalla.

Olisi anteeksiantamatonta unohtaa Laura Sackerman ja Marc Young, jotka taidokkaasti toimittivat tämän kirjan edellisen painoksen. Heidän loistavan työnsä ansiosta uuden painoksen laatiminen oli lasten leikkiä.

Michaelin suosikkipaikat Webissä

Kokeile näitä Web-paikkoja, kun etsit lisätietoja Visual Basicista!

http://www.apexsc.com/vb/ ***Carl and Gary***

Carlin ja Garyn Visual Basic Home Page on ehkä kiehtovin ja kattavin "omista" VB-kotisivuista. Siihen kannattaa ehdottomasti tutustua, koska siinä on laaja kokoelma linkkejä ja resursseja.

http://www.microsoft.com/vbasic/ ***Microsoft Visual Basic***

Tämä on Microsoft Corporationin virallinen Visual Basic -sivu. Siitä löydät uutisia, lähdeohjelmia, teknisiä julkaisuja sekä mukautettuja sivuja, jotka on tarkoitettu aloittelijoille, kokeneille ohjelmoijille ja Web-ohjelmoijille.

http://www.microsoft.com/mspress/ ***Microsoft Press***

Microsoft Pressin kotisivu, josta löydät tiedot Microsoft Pressin uusimmista Visual Basic -kirjoista. Sen kautta voit myös kopioida ilmaistuotteita ja lähettää postia Microsoft Pressille.

http://www.microsoft.com/officedev/ ***Visual Basic for Applications***

Microsoft Office Developer Forum sisältää tietoja, vihjeitä ja palveluita VB-ohjelmoijille, jotka kirjoittavat makroja Microsoft Office -sovelluksiin.

http://web2.airmail.net/gbeene/ ***Gary Beene's World***

Gary Beenen Visual Basic World on lämminhenkinen Web-paikka, jossa on paljon hyödyllisiä linkkejä ja tietoja uusille ohjelmoijille.

http://www.citilink.com/~jgarrick/vbasic/ ***Joe Garrick***

Joe Garrickin Visual Basic -kotisivulla on lähdeohjelmia, vihjeitä, Web-linkkejä ja yleisiä tietoja VB-ammattilaisille.

http://crescent.progress.com/ ***Crescent Software***

Crescent on Visual Basic -työkalujen valmistaja, joka kuuluu Progress Softwareen. Tästä Web-paikasta löydät hyödyllisiä, kaupallisia Visual Basic -lisäohjelmia, esimerkiksi ActiveX-objekteja ja Web-työkaluja.

http://home.sprynet.com/sprynet/rasanen/vbnet/default.htm ***VBNet***

VBNet on pieni ja mukava kotisivu, joka sisältää paljon läpikotaisin mietittyjä malliohjelmia kokeneille Visual Basic -kehittäjille. Se käsittelee muun muassa yleisiä valintaikkunoita, bittikarttojen API:tä ja rekisteritietoja.

http://www.netgate.net/~vbshop/vb.html ***VBShop***

Visual Basic Shop on kiva valikoima ilmaisia työkaluja ja vihjeitä Visual Basic -ohjelmoijille.

http://www.inquiry.com/ ***Inquiry Search Tools***

Inquiry on ilmaispalvelu tietotekniikan ammattilaisille. Tämän linkin kautta pääset tietokantaan, jossa on 130 000 tietotekniikan julkaisua, eri valmistajien tuotteita ja teknisiä vihjeitä. Kokeile hakusanaa Visual Basic ja katso, montako artikkelia löydät! Kokeile myös osaa Visual Basic Pro Ask Me.

Tämä luettelo on laadittu tammikuussa 1997. Koska useimpien Web-paikkojen linkit vaihtuvat vähän väliä, osa yllä mainituista osoitteista saattaa olla vanhentuneita tai pois käytöstä. Jos sinulla on Microsoft Word 97, voit avata tämän luettelon elektronisen version (Web-linkkeineen) oheisen CD-tietolevyn päähakemiston tiedostosta Top10.doc.

Sisällysluettelo

Pikaopas xi

Sopivan aloituskohdan valitseminen xiii

Itsellesi sopivan aloituskohdan valitseminen xiv ▯ Visual Basic 5:n uusia ominaisuuksia xv ▯ Korjaukset, kommentit ja ideat xvi ▯ Vieraile Web-paikassamme xvi

Harjoitustiedostojen asentaminen ja käyttö xvii

Harjoitustiedostojen käyttö xix ▯ Harjoitustiedostojen poistaminen xxiii ▯ Ongelmia harjoitustiedostoissa? xxiii

Tämän kirjan merkintätapoja ja lukemisohjeita xxv

Merkintätapoja xxv ▯ Lukemisohjeita xxvi

Osa 1 Tutustuminen Visual Basiciin

Harjoitus 1 Ensimmäisen ohjelman kirjoittaminen 3

Visual Basic -ohjelmointiympäristö 3 ▯ Onnen seiska: ensimmäinen Visual Basic -ohjelmasi 15 ▯ Lisätietoja: Lisäysten tekeminen ohjelmaan 37 ▯ Harjoituksen tiivistelmä 39 ▯ Tietoja seuraavasta harjoituksesta 41

Harjoitus 2 Objektien käyttö 43

Objektien alkeita: "Hello World" -ohjelma 44 ▯ Tiedostojärjestelmäobjektien käyttö 48 ▯ Syöttötietoja keräävät objektit 54 ▯ Sovellusten käynnistäminen OLE-objektilla 60 ▯ Microsoft Access -tietokannan katseleminen dataobjektin avulla 66 ▯ Tietokannan muokkaaminen 71
▯ Lisätietoja: ActiveX-objektien asentaminen 71 ▯ Harjoituksen tiivistelmä 75 ▯ Tietoja seuraavasta harjoituksesta 76

Harjoitus 3 Valikoiden ja valintaikkunoiden käyttö 77

Valikoiden lisääminen Menu Editorilla 77 ▯ Valikkokomentojen käsittely 83 ▯ Yleisten valintaikkunaobjektien käyttö 88 ▯ Yleisiä valintaikkunoita ohjaavat tapahtuma-aliohjelmat 94 ▯ Lisätietoja: Pikanäppänten lisääminen valikoihin 101 ▯ Harjoituksen tiivistelmä 103 ▯ Tietoja seuraavasta harjoituksesta 104

Osa 2 Ohjelmoinnin perusteita

Harjoitus 4 Visual Basicin muuttujat ja operaattorit 107

Visual Basic -ohjelmalauseen rakenne 108 ▯ Tietojen tallentaminen muuttujiin 108 ▯ Syöttötietojen tallentaminen muutujaan 113 ▯ Tietojen näyttäminen muuttujalla 116 ▯ Tietotyyppien käyttö 118 ▯ Visual Basicin operaattorien käyttö 123 ▯ Lisätietoja: Sulkeiden käyttö kaavassa 131 ▯ Harjoituksen tiivistelmä 132 ▯ Tietoja seuraavasta harjoituksesta 133

Harjoitus 5 Ehtorakenteiden käyttö 135

Tapahtumaohjattu ohjelmointi 135 ▯ Ehtolausekkeiden käyttö 137 ▯ If…Then-ehtorakenteet 138 ▯ Select Case -ehtorakenteet 145 ▯ Virheiden etsiminen ja korjaaminen 150 ▯ Lisätietoja: Askellustilan valitseminen Stop-lauseella 157 ▯ Harjoituksen tiivistelmä 158 ▯ Tietoja seuraavasta harjoituksesta 159

Harjoitus 6 Silmukoiden ja ajastimien käyttö 161

For…Next-silmukoiden kirjoittaminen 161 ▯ Do-silmukoiden kirjoittaminen 173 ▯ Ajastinobjektien käyttö 176 ▯ Lisätietoja: Aikarajan määrittäminen ajastinobjektilla 179 ▯ Harjoituksen tiivistelmä 182 ▯ Tietoja seuraavasta harjoituksesta 184

Osa 3 Käyttöliittymän luominen

Harjoitus 7 Lomakkeet, tulostimet ja virheenkäsittelijät 187

Uusien lomakkeiden lisääminen ohjelmaan 187 ▯ Ohjelman tietojen lähettäminen tulostimeen 196 ▯ Virheisiin varautuminen virheenkäsittelijöillä 203 ▯ Lisätietoja: Muita tapoja käsitellä virheet 208 ▯ Harjoituksen tiivistelmä 210 ▯ Tietoja seuraavasta harjoituksesta 212

Harjoitus 8 Kuvioiden ja erikoistehosteiden lisääminen 213

Kuvien lisääminen käyttämällä Viiva- ja Kuvio-työkaluja 214 ▯ Graafisten komentopainikkeiden luominen 219 ▯ Vedon ja pudotuksen tuen lisääminen ohjelmiin 227 ▯ Animaation lisääminen ohjelmaan 234 ▯ Lisätietoja: Ohjelman objektien nimeäminen 245 ▯ Harjoituksen tiivistelmä 210 ▯ Tietoja seuraavasta harjoituksesta 246

Osa 4 Laajemmat ohjelmat

Harjoitus 9 Moduulien ja aliohjelmien käyttö 249

Vakiomoduulien käsittely 249 ▯ Julkisten muuttujien käsittely 253 ▯ Yleiskäyttöisten aliohjelmien esittelemi-nen 257 ▯ Funktioaliohjelmien kirjoittaminen 258 ▯ Lausealiohjelmien kirjoittaminen 264 ▯ Lisätietoja: Arvoargumentit 271 ▯ Harjoituksen tiivistelmä 272 ▯ Tietoja seuraavasta harjoituksesta 274

Harjoitus 10 Kokoelmien ja taulukoiden käsittely 275

Objektikokoelmien käsittely 275 ▯ Muuttujataulukoiden käsittely 282 ▯ Lisätietoja: Moniulotteisten taulukoiden käyttö 293 ▯ Harjoituksen tiivistelmä 297 ▯ Tietoja seuraavasta harjoituksesta 299

Harjoitus 11 Tekstitiedostojen ja tietokantojen käsittely 301

Tekstitiedoston sisällön näyttäminen tekstiruutuobjektilla 302 ▯ Tekstitiedoston luominen levylle 307 ▯ Tietokantojen hallinta 312 ▯ Recordset-objektin käyttö 316 ▯ Tietueiden lisääminen Biblio.mdb-tietokantaan 319 ▯ Tietueiden poistaminen Biblio.mdb-tietokannasta 322 ▯ Lisätietoja: Tiedoston varmistus-kopiointi 325 ▯ Harjoituksen tiivistelmä 327 ▯ Tietoja seuraavasta harjoituksesta 329

Harjoitus 12 Yhteydet Microsoft Officeen 331

Yrityksen tietojärjestelmän luominen 331 ▯ Sovellusobjektien ohjelmoiminen Automation-tekniikalla 341 ▯ Lisätietoja: Etsintä ja korvaus käyttämällä Wordia 351 ▯ Harjoituksen tiivistelmä 353 ▯ Mitä seuraavaksi? 354

Liite Ajatteleminen ohjelmoijan tavalla 355

Mikä on ohjelma? 355 ▯ Ohjelman testaaminen, kääntäminen ja jakeleminen 358

Hakemisto 361

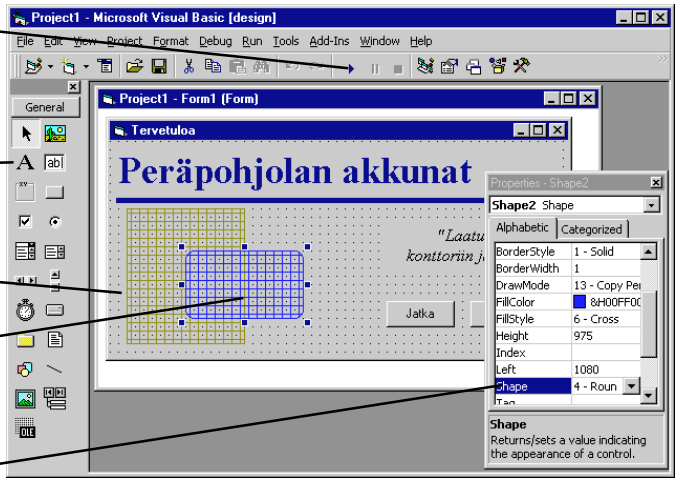
Ohjelman suorittaminen ja Windows 95:ssä suoritettavan tiedoston luominen
katso sivun 35 kohtaa Suoritettavan tiedoston muodostaminen (harjoitus 1)

Objektien luominen käyttöliittymän suunnittelutyökaluilla
katso sivun 9 kohtaa Työkaluruutu (harjoitus 2)

Käyttöliittymän luominen
katso sivun 8 kohtaa Käyttöliittymälomake (harjoitus 1)

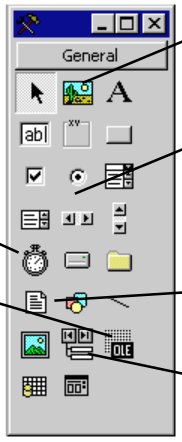
Piirrosten lisääminen
katso sivun 214 kohtaa Kuvio-työkalu (harjoitus 8)

Objektien ominaisuuksien määrittäminen
katso sivun 9 kohtaa Properties-ikkuna (harjoitus 1)



Kellojen ja ajastimien luominen
katso sivun 176 kohtaa Ajastinobjektien käyttö (harjoitus 6)

Microsoft Office -sovellusten ohjaaminen
katso sivun 333 kohtaa OLE-työkalun käyttö (harjoitus 12)

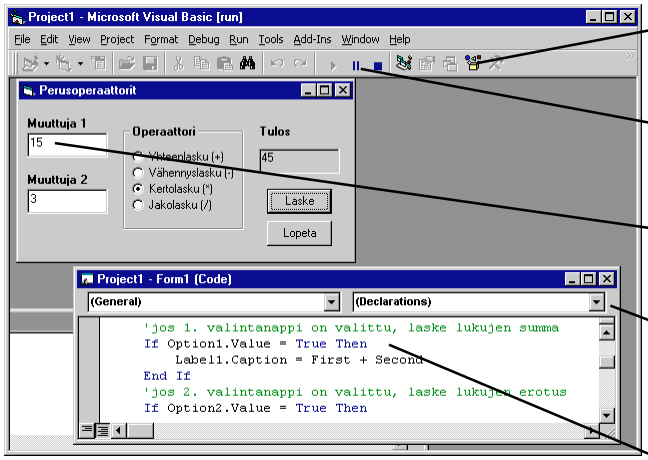


Kuvien ja animaatioiden lisääminen
katso sivun 234 kohtaa Animaation lisääminen ohjelmaan (harjoitus 8)

Omien lomakkeiden ja valintaikkunoiden luominen
katso sivun 54 kohtaa Syöttötietoja keräävät objektit (harjoitus 2)

Tiedostoselaimen luominen
katso sivun 48 kohtaa Tiedosto-järjestelmäobjektien käyttö (harjoitus 2)

Tietokantojen katseleminen ja muokkaaminen
katso sivun 312 kohtaa Tietokantojen hallinta (harjoitus 11)



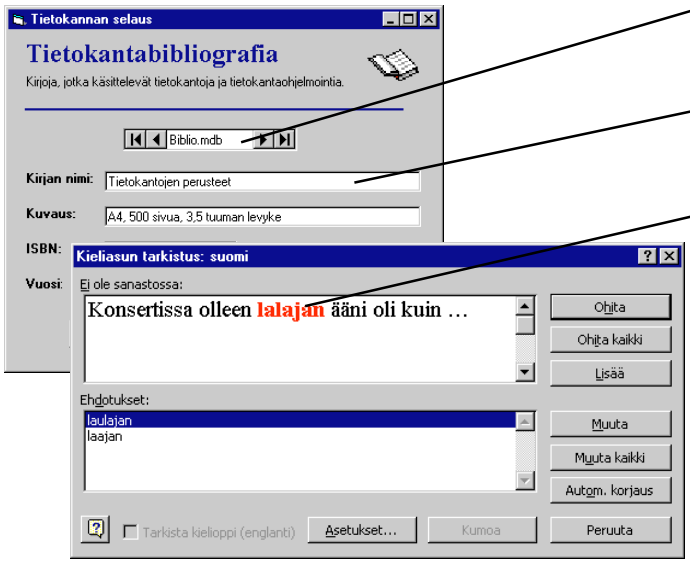
Ohjelman objektien tutkiminen
katso sivun 343 kohtaa Visual Basic Object Browser (harjoitus 12)

Vianetsintä
katso sivun 150 kohtaa Virheiden etsiminen ja korjaaminen (harjoitus 5)

Tietojen käsitteleminen ohjelmassa
katso sivun 108 kohtaa Tietojen tallentaminen muuttujiin (harjoitus 4)

Visual Basic -ohjelmalauseiden kirjoittaminen ja muokkaaminen
katso sivun 27 kohtaa Ohjelmalauseiden kirjoittaminen (harjoitus 1)

Ohjelman logiikan muodostaminen
katso sivun 138 kohtaa If...Then-ehtorakenteet (harjoitus 5)



Dataobjektin käyttö
katso sivun 312 kohtaa Tietokantojen hallinta (harjoitus 11)

Tietokannan kenttien näyttäminen
katso sivun 313 kohtaa Tietojen näyttäminen tekstiruutuobjekteilla (harjoitus 11)

Tekstiruudun oikeinkirjoituksen tarkistaminen Microsoft Wordilla
katso sivun 341 kohtaa Sovellusobjektien ohjelmoiminen Automation-tekniikalla (harjoitus 12)

Sopivan aloitus-kohdan valitseminen

Microsoft Visual Basic 5.0 on tehokas ohjelmointijärjestelmä, jolla rakennetaan nopeasti ja tehokkaasti sovelluksia Microsoft Windows 95:een tai Microsoft Windows NT:hen. *Opeta itsellesi Microsoft Visual Basic 5* näyttää, miten opit jo muutamassa tunnissa kirjoittamaan omia ohjelmia Visual Basicilla. Siinä esitetyt tiedot koskevat Visual Basic Learning Editionia, Visual Basic Professional Editionia ja Visual Basic Enterprise Editionia.

TÄRKEÄÄ Tämä kirja on tarkoitettu Microsoft Visual Basic 5.0:lle sekä Windows 95- tai Windows NT 4.0 -käyttöjärjestelmälle. Jos haluat tietää, mikä Visual Basicin versio sinulla on, katso ohjelmiston pakkausta tai käynnistä ohjelmisto ja valitse Help-valikosta komento About Microsoft Visual Basic. Ellei ohjelmistosi ole yhteensopiva tämän kirjan kanssa, löydät kirjakaupasta useita muita Step By Step -sarjan kirjoja. Voit myös kokeilla Microsoft Pressin Web-paikkaa, jonka osoite on http://www.microsoft.com/mspress/.

Itsellesi sopivan aloituskohdan valitseminen

Tämä kirja on tarkoitettu lukijoille, jotka opettelevat Microsoft Visual Basicia alusta alkaen, siirtyvät Visual Basiciin jostakin muusta ohjelmointikielestä tai jotka tuntevat Visual Basicin ja haluavat oppia Visual Basic 5.0:n uudet ominaisuudet. Etsi itsellesi sopiva aloituskohta seuraavan taulukon avulla.

Jos olet	Toimi näin
Uusi...	
ohjelmoija	1. Asenna harjoitustiedostot kohdassa "Harjoitustiedostojen asentaminen ja käyttö" kuvatulla tavalla. 2. Tutustu ohjelmointiprosessin perusvaiheisiin lukemalla liite "Ajatteleminen ohjelmoijan tavalla". 3. Opettele Microsoft Visual Basicin perusteet tekemällä harjoitukset 1–12.
Jos olet	**Toimi näin**
Ohjelmoinut...	
Visual Basic for Applicationsilla	1. Asenna harjoitustiedostot kohdassa "Harjoitustiedostojen asentaminen ja käyttö" kuvatulla tavalla". 2. Tee osan 1 harjoitukset, lue läpi osa 2 ja tee osien 3 ja 4 harjoitukset.
Jos olet	**Toimi näin**
Ohjelmoinut...	
Microsoft QuickBasicilla, MS-DOS QBasicilla tai C:llä	1. Asenna harjoitustiedostot kohdassa "Harjoitustiedostojen asentaminen ja käyttö" kuvatulla tavalla. 2. Tee harjoitus 1, lue läpi harjoitukset 2–6, ja tee osien 3 ja 4 harjoitukset. 3. Löydät ohjeet käyttöliittymän luomisesta Visual Basicilla harjoituksista 2, 3, 7 ja 8.

Jos olet	Toimi näin
Kertaamassa...	
tätä kirjaa tehtyäsi kaikki harjoitukset	1. Katso hakemistosta, missä kirjan kohdissa käsitellään kiinnostavia aiheita, tai etsi yleisiä ohjeita sisällysluettelon ja Pikaoppaan avulla. 2. Lue kunkin harjoituksen lopusta "Harjoituksen tiivistelmä", johon on koottu harjoituksen olennaiset tiedot. Ne ovat samassa järjestyksessä kuin ne on käsitelty harjoituksessa.

Visual Basic 5:n uusia ominaisuuksia

Alla on lueteltu Microsoft Visual Basicin tärkeimmät uudet ominaisuudet, joita käsitellään tässä kirjassa. Taulukossa on mainittu harjoitukset, joissa kutakin ominaisuutta kuvataan. Löydät toimintoa tai tehtävää koskevia ohjeita myös hakemistosta.

Jos haluat oppia	Katso
Hallitsemaan projekteja Project Explorerilla	Harjoitus 1
Käyttämään editorin uusia toimintoja	Harjoitus 1
Hienosäätämään lomakkeiden ulkoasua Form Layout -ikkunan avulla	Harjoitus 1
Pienentämään, suurentamaan ja telakoimaan Visual Basic -ympäristön ikkunoita ja työkaluja	Harjoitus 1
Valitsemaan ja käyttämään ActiveX-objekteja	Harjoitus 2
Korjaamaan ohjelman virheitä uusilla vianetsintätyökaluilla ja -menetelmillä	Harjoitus 5
Luomaan projektin käyttämällä esimääritettyä lomaketta	Harjoitus 7
Käyttämään muiden Microsoft-sovellusten toimintoja Automation-tekniikalla	Harjoitus 12

Korjaukset, kommentit ja ideat

Tämän kirjan sisältö ja harjoitustiedostot on pyritty tekemään mahdollisimman virheettömiksi. Microsoft Press ilmoittaa kirjoihinsa tehdyistä korjauksista ja lisäyksistä Word Wide Webin kautta osoitteessa

http://www.microsoft.com/mspress/support/

Jos sinulla on kommentteja, kysymyksiä tai ideoita tästä kirjasta tai sen harjoitustiedostoista, lähetä ne Microsoft Pressille.

Lähetä sähköpostia osoitteella

mspinput@microsoft.com

tai perinteistä postia osoitteella

Microsoft Press
Attn: Step by Step Series Editor
One Microsoft Way
Redmond, WA 98052-6399
USA

Huomaa, että nämä osoitteet eivät ole Visual Basic -ohjelmistotuotteiden teknistä tukea varten.

Vieraile Web-paikassamme

Olet tervetullut vierailemaan Microsoft Pressin World Wide Web -paikassa. Sen osoite on

http://www.microsoft.com/mspress/

Siitä löydät Microsoft Pressin kaikkien kirjojen esittelyt, tietoja kirjojen tilaamisesta, ilmoituksia erikoisominaisuuksista ja tapahtumista, lisäyksiä Microsoft Pressin kirjoihin ja paljon muuta.

Löydät tuoreimmat uutiset Microsoft Corporationista ja sen ohjelmatuotteista vierailemalla seuraavassa Web-paikassa:

http://www.microsoft.com/

Harjoitustiedostojen asentaminen ja käyttö

Tämän kirjan takakannen suojataskussa on CD-tietolevy, jolta löydät kirjan harjoituksissa tarvitsemasi tiedostot. Kun esimerkiksi kokeilet tietokannan käsittelemistä Visual Basicilla, avaa yksi harjoitustiedostoista — tietokanta nimeltä Shops.md — ja muokkaa sitä Visual Basicin dataobjektilla. Kun sinulla on valmiit harjoitustiedostot, sinun ei tarvitse kuluttaa aikaa harjoituksissa kuvattujen ohjelmien kirjoittamiseen. Voit sen sijaan keskittyä Visual Basicin ohjelmointitekniikkojen omaksumiseen. Kun käytät harjoitustiedostoja ja noudatat harjoitusten vaiheittaisia ohjeita, opit tekemällä, mikä on helppo ja tehokas tapa omaksua ja painaa mieleen uusia taitoja.

TÄRKEÄÄ Ennen kuin avaat harjoituslevyn suojataskun sinetin, varmista, että sinulla on tälle kirjalle sopiva Visual Basicin versio. Tämä kirja on tarkoitettu Microsoft Visual Basic 5.0:lle sekä Windows 95- tai Windows NT 4.0 -käyttöjärjestelmälle. Jos haluat tietää, mikä Visual Basicin versio sinulla on, katso ohjelmiston pakkausta tai käynnistä ohjelmisto ja valitse Help-valikosta komento About Microsoft Visual Basic. Ellei ohjelmistosi ole yhteensopiva tämän kirjan kanssa, löydät kirjakaupasta useita muita Step by Step -sarjan kirjoja. Voit myös kokeilla Microsoft Pressin Web-paikkaa, jonka osoite on http://www.microsoft.com/mspress/.

Harjoitustiedostojen asentaminen tietokoneeseen

Asenna harjoitustiedostot tietokoneesi kiintolevylle seuraavien ohjeiden mukaisesti, jotta voit käyttää niitä kirjan harjoitusten yhteydessä.

1. Ellei tietokoneesi ole käynnissä, kytke siihen virta.
2. Jos sinulla on Windows NT, paina CTRL+ALT+DEL, jotta se kysyy käyttäjätunnuksesi ja salasanasi. Windows 95 kysyy samat tiedot, jos tietokoneesi on yhteydessä verkkoon.

Jos sinulla on Windows 95 ja tietokoneeseesi on määritetty käyttäjäprofiilit, järjestelmä kysyy käyttäjätunnuksen ja salasanan käynnistyksen jälkeen.

3. Kirjoita käyttäjätunnuksesi ja salasanasi tekstiruutuihin ja napsauta OK. Jos näyttöön tulee Tervetuloa-valintaikkuna (Welcome), napsauta Sulje-painiketta (Close).
4. Ota harjoituslevy tämän kirjan takakannen taskusta.
5. Aseta levy CD-levyasemaan.
6. Valitse näytön alareunan tehtäväpalkista Käynnistä (Start) ja Suorita (Run).

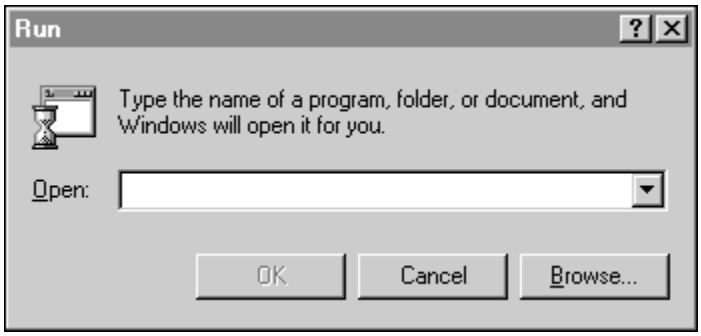
Ellet tiedä käyttäjätunnustasi tai salasanaasi, kysy neuvoa järjestelmän pääkäyttäjältä.

...and then
... ja sitten Suorita (Run)

Click Start...
Napsauta Käynnistä (Start)...

Suorita-valintaikkuna (Run) avautuu.

7. Kirjoita tekstiruutuun **d:setup**. Jos CD-levyasemasi tunnuskirjain on jokin muu, esimerkiksi e:, kirjoita se komennon alkuun.



8 Napsauta OK ja noudata näyttöön tulevia ohjeita.

Asennusohjelman ikkuna avautuu, ja suositeltavat asetukset on valittu valmiiksi. Harjoitustiedostot toimivat parhaiten, jos hyväksyt oletusasetukset. (Jos sijoitat tiedostot johonkin toiseen kansioon, sinun pitää muuttaa eräissä tiedostoissa olevia polkunimiä, jotta ohjelmat löytävät tietyt välttämättömät osat, kuten piirrokset ja tietokannat.)

9 Kun harjoitustiedostot on asennettu, ota levy CD-levyasemasta ja aseta se takaisin takakannen taskuun.

Kiintolevyllä on nyt kansio nimeltä \Vb5Sbs. Se sisältää tämän kirjan harjoitustiedostot.

HUOMAA Samalla kun asennusohjelma kopioi harjoitustiedostot, se lisää Windowsin työpöydälle pikakuvakkeen, jolla saat yhteyden Microsoft Pressin Web-paikkaan. Jos tietokoneessasi on Internet-yhteys, pikakuvaketta napsauttamalla löydät Microsoft Pressin Web-paikan. Voit myös käyttää osoitetta http://www.microsoft.com/mspress/.

Harjoitustiedostojen käyttö

Tämän kirjan jokaisessa harjoituksessa kerrotaan, milloin ja miten siihen liittyviä harjoitustiedostoja käytetään. Kun tarvitset harjoitustiedostoa, kerron sinulle, miten voit avata sen. Harjoitukset jäljittelevät todellisia ohjelmointiprojekteja, joten voit helposti soveltaa oppimiasi taitoja omaan työhösi.

Jos haluat tietää kaikki yksityiskohdat, katso seuraavasta luettelosta, mitä Visual Basic -projektitiedostoja (.vbp) harjoituslevyllä on:

Projekti	Kuvaus
Harjoitus 1	
StepUp	Yksinkertainen animaatio-ohjelma, joka toivottaa sinut tervetulleeksi ohjelmointikurssille.
Lucky	Ensimmäinen ohjelmasi — onnenpeli, joka simuloi peliluolista tuttua yksikätistä rosvoa.
Harjoitus 2	
Hello	”Hello, World!” -ohjelma, joka havainnollistaa teksti- ja tekstiruutuobjektien käyttämistä.
Online	Verkkokauppaan tarkoitetun ohjelman käyttöliittymä, jossa on useita käteviä tiedonsyöttöobjekteja.
Browser	Bittikartan katseluohjelma, jonka tiedostojärjestelmä-objekteilla avataan kuvatiedostoja.

Projekti	Kuvaus
Data	Yksinkertainen tietokannan katseluohjelma, joka esittelee dataobjektia.
OleBid	Projektin suunnitteluväline, joka käynnistää Windows-sovelluksia OLE-objektilla.
Harjoitus 3	
Menu	Näyttää, miten lomakkeeseen lisätään valikoita ja komentoja.
Dialog	Muuttaa lomakkeen tekstin värin yleisellä valintaikkuna-objektilla.
Harjoitus 4	
VarTest	Esittelee muunnelmatyyppisiä muuttujia ja tallentaa niihin tietoja.
MsgBox	Näyttää tietoja MsgBox-funktiolla.
InputBox	Lukee tietoja InputBox-funktiolla.
Data	Esittelee perustietotyyppejä.
Constant	Säilyttää lukuarvon vakiossa.
BasicOp	Kuvaa yhteen-, vähennys-, kerto- ja jakolaskuoperaattoreiden käyttöä.
AdvOp	Kuvaa kokonaislukujako-, jakojäännös-, potenssiinkorotus- ja yhteenliittämisoperaattoreiden käyttöä.
Harjoitus 5	
Login	Ohjaa järjestelmään kirjautumista If…Then…Else-rakenteella.
Pass	Tarkistaa kirjautumissalasanan loogisella And-operaattorilla.
Case	Valitsee Case-rakenteella tervetulotoivotuksen, joka on kirjoitettu halutulla kielellä.
IfBug	Virheenetsintäharjoitus (löydätkö loogisen virheen?)
Harjoitus 6	
ForLoop	Tulostaa useita rivejä For...Next-silmukassa.
GrowFont	Muuttaa FontSize-ominaisuutta silmukassa.
CtlArray	Avaa tiedostoja käyttäen silmukkaa ja objektitaulukkoa.
StepLoop	Näyttää kuvakkeita käyttäen Step-avainsanaa.
Celcius	Muuntaa lämpötiloja Do-silmukassa.
DigClock	Digitaalikello.
TimePass	Kirjautumisohjelma, jossa on salasanalle aikaraja.

Projekti	Kuvaus
Harjoitus 7	
Italian	Italian sanakirja, joka näyttää verbien käännöksiä MsgBox-funktiolla.
Italian2	Italian sanakirja, joka näyttää verbien käännöksiä toisessa lomakkeessa.
PrintFrm	Ohjelma, joka lähettää muotoillun tekstin tulostimeen.
PrintWMF	Ohjelma, joka tulostaa Windows-metatiedoston (.wmf).
DriveErr	Ohjelma, joka kaatuu, jos levykeasemaa ei käytetä oikein (vain kokeilutarkoituksiin).
FinalErr	Ohjelma, joka käsittelee levykeaseman virheen.
Harjoitus 8	
StartFrm	Aloituslomake, joka kuvaa Viiva- ja Kuvio-työkalujen käyttämistä.
Buttons	Ohjelma, joka kuvaa graafisten komentopainikkeiden suunnittelemista ja käyttämistä.
DragDrop	Ohjelma, jossa vedetään ja pudotetaan kuvakkeita polttotynnyriin.
Smoke	Ohjelma, joka näyttää leijuvaa savupilveä esittävän animaation.
Zoom	Ohjelma, joka simuloi maata lähestyvää avaruusalusta.
NameConv	Ohjelma, joka esittelee objektien nimeämissääntöjä.
Harjoitus 9	
Wins	Ohjelma, joka säilyttää onnenpelin voittojen määrän julkisessa muuttujassa.
Rate	Ohjelma, joka laskee onnenpelin voittosuhteen.
Teams	Yleiskäyttöinen lausealiohjelma, joka lisää vaihtoehtoja luetteloruutuun.
Harjoitus 10	
Move	Siirtää objekteja lomakkeessa kokoelmana.
Tag	Käsittelee kokoelman yhden objektin muista poikkeavalla tavalla käyttäen Tag-ominaisuutta.
FixArray	Laskee vakiokokoiseen taulukkoon tallennettujen lämpötilojen keskiarvon.

Projekti	Kuvaus
DynArray	Laskee dynaamisesti muuttuvaan taulukkoon tallennettujen lämpötilojen keskiarvon.
Baseball	Tallentaa baseball-ottelun tulokset kaksiulotteiseen taulukkoon.
Harjoitus 11	
ShowText	Näyttää tekstitiedoston sisällön.
Qnote	Yksinkertainen muistiinpanojen kirjoitusohjelma.
DataView	Muokkausohjelma Biblio.mdb-tietokannalle (sisältyy Visual Basiciin).
FindData	Ohjelma, jolla etsitään tietoja tietokannan kentistä.
AddData	Ohjelma, joka lisää tietueen tietokantaan.
DelData	Ohjelma, joka poistaa tietueen tietokannasta.
Backup	Ohjelma, joka tekee tietokannasta kopion ennen avaamista.
Harjoitus 12	
MusicEIS	Tietojärjestelmäsovellus, joka käyttää Excel-taulukkoa, Excel-taulukkoa, kuvatiedostoa ja Access-tietokantaa.
UseWord	Ohjelma, joka tarkistaa tekstiruudun sisällön oikeinkirjoituksen kutsumalla Wordia.
WdSearch	Ohjelma, joka etsii ja korvaa tekstiruudun tekstiä kutsumalla Wordia.
Kertaustehtävät	
Rev1.doc	Osan 1 ohjelmointiharjoitus, jossa tehdään bittikarttojen selausohjelma.
Rev2.doc	Osan 2 ohjelmointiharjoitus, jossa tehdään tapaamisista muistuttava kello.
Rev3.doc	Osan 3 ohjelmointiharjoitus, jossa tehdään parannettu bittikarttojen selausohjelma, jossa on vedon ja pudotuksen sekä tulostuksen tuki.
Rev4.doc	Osan 4 ohjelmointiharjoitus, jossa tehdään laajempi tietokannan katseluohjelma.

Harjoitustiedostojen poistaminen

Noudata seuraavia ohjeita, jos haluat poistaa tämän kirjan harjoitustiedostot kiintolevyltä.

1. Valitse Käynnistä (Start), Asetukset (Settings) ja Ohjauspaneeli (Control Panel).
2. Kaksoisnapsauta Lisää/poista sovellus -kuvaketta (Add/Remove Programs).
3. Valitse luettelosta Microsoft Visual Basic 5 Step by Step ja napsauta Lisää/Poista (Add/Remove).

   Järjestelmä pyytää vahvistuksen.
4. Napsauta Kyllä (Yes).

   Järjestelmä poistaa harjoitustiedostot.
5. Sulje valintaikkuna napsauttamalla OK.
6. Sulje Ohjauspaneelin ikkuna.

Ongelmia harjoitustiedostoissa?

Tämän kirjan sisältö ja harjoitustiedostot on pyritty tekemään mahdollisimman virheettömiksi. Jos kuitenkin havaitset ongelmia, löydät ehkä korjatut tiedostot Microsoft Pressin Web-paikasta, jonka osoite on

http://www.microsoft.com/mspress/support/

Voit myös vierailla kotisivullamme, joka on osoite on

http://www.microsoft.com/mspress/

Tältä sivulta löydät Microsoft Pressin kaikkien kirjojen esittelyt, tietoja kirjojen tilaamisesta, ilmoituksia erikoisominaisuuksista ja tapahtumista, lisäyksiä Microsoft Pressin kirjoihin ja paljon muuta.

Tämän kirjan merkintätapoja ja lukemisohjeita

Sinun kannattaa ennen harjoitusten aloittamista tietää, miten ohjeet, painettavat näppäimet ja muut seikat on esitetty tässä kirjassa. Lue seuraava luettelo ja lukemisohjeet, jotta osaat käyttää tätä kirjaa mahdollisimman tehokkaasti.

Merkintätapoja

- Käytännön tehtävien toimintaohjeet, joita sinun tulee noudattaa, on esitetty numeroituina vaiheittaisina luetteloina (1, 2 ja niin edelleen). Nuoli (➤) tarkoittaa tehtävää, jossa on vain yksi vaihe.

- Teksti, joka sinun pitää kirjoittaa, on **lihavoitu**.
- Plusmerkki (+) kahden näppäimen nimen välissä tarkoittaa, että sinun on painettava näppäimiä yhtaikaa. Esimerkiksi ”Paina ALT+SARKAIN” merkitsee ALT-näppäimen pitämistä alhaalla ja SARKAINNÄPPÄIMEN painamista.

- Kappaleissa, joissa on merkintä HUOMAA tai VIHJE tai jotka on sijoitettu vasempaan reunukseen, on lisätietoja tai vaihtoehtoinen menettelytapa.

- Kappaleet, joissa on merkintä TÄRKEÄÄ tai VAROITUS, kehottavat tarkistamaan tietyn asian ennen harjoituksen jatkamista.

Lukemisohjeita

- Harjoituksissa on varjostettuja tekstikohtia, joissa on ohjelmoinnin erikoismenetelmien kuvauksia, taustatietoja ja tietoja käsiteltävään aiheeseen liittyvistä piirteistä. Ne selventävät terminologiaa tai ehdottavat, millaisiin toimintoihin sinun kannattaa tutustua.
- Jos haluat perehtyä tarkemmin harjoituksessa esitettyihin toimintoihin tai menetelmiin, kokeile harjoituksen lopussa olevaa tehtävää, joka alkaa otsikolla Lisätietoja.
- Voit nopeasti kerrata harjoituksessa esitetyt tehtävät lukemalla tiivistelmän, joka sijaitsee harjoituksen lopussa.
- Jokaisen harjoituksen lopussa on luettelo Visual Basicin ohjeiden aiheista, joissa on lisätietoja. Ohjejärjestelmässä on täydellinen Microsoft Visual Basicin käyttöopas elektronisessa muodossa. Ohjejärjestelmää kuvataan harjoituksessa 1.
- Jos haluat kerrata tehtävissä esitettyjä asioita, katso oheisen CD-tietolevyn kansiota \Vb5Sbs\Review. Siinä on neljä harjoitusta, jotka liittyvät tämän kirjan eri osiin. Jokainen sisältää ”todenmukaisen” ohjelmointitehtävän, joka sisältää syventäviä tietoja ja havainnollistaa tehokkaasti tapoja, joilla voit soveltaa oppimaasi.

Osa

1

Tutustuminen Visual Basiciin

Harjoitus 1
Ensimmäisen ohjelman kirjoittaminen 3

Harjoitus 2
Objektien käyttö 43

Harjoitus 3
Valikoiden ja valintaikkunoiden käyttö 77

Harjoitus

1

Ensimmäisen ohjelman kirjoittaminen

Arvioitu aika
55 min.

Tässä harjoituksessa opit

- käynnistämään Visual Basicin
- käyttämään Visual Basic -ohjelmointiympäristöä
- kirjoittamaan ja suorittamaan ensimmäisen Visual Basic -ohjelmasi
- tallentamaan ohjelman ja lopettamaan Visual Basicin.

Tässä ja seuraavissa harjoituksissa huomaat, miten helppoa Microsoft Visual Basicilla on kirjoittaa tehokkaita Microsoft Windows -perustaisia ohjelmia. Visual Basicin kyky suorittaa vaikuttavia tehtäviä perustuu avainkäsitteisiin, joita kuvaan tarkasti kirjan ensimmäisessä osassa. Vaikka et olisi aikaisemmin kirjoittanut ohjelmia, näet pian, että tarvitset ohjelmoinnissa hyvin samantapaista järkeilyä ja tietokonetaitoja kuin jokapäiväisessä työssäsi. Tässä harjoituksessa kerron, miten Visual Basic käynnistetään ja miten Visual Basic -ohjelmointijärjestelmällä kirjoitetaan ja suoritetaan ohjelmia. Opit Visual Basicin tärkeimmät valikkokomennot ja ohjelmointitoiminnot, suoritat yksinkertaisen Visual Basic -ohjelman nimeltä StepUp, ja laadit toisen ohjelman nimeltä Lucky Seven. Opit myös muokkaamaan ohjelmaa, tallentamaan sen levylle ja lopettamaan Visual Basicin.

Visual Basic -ohjelmointiympäristö

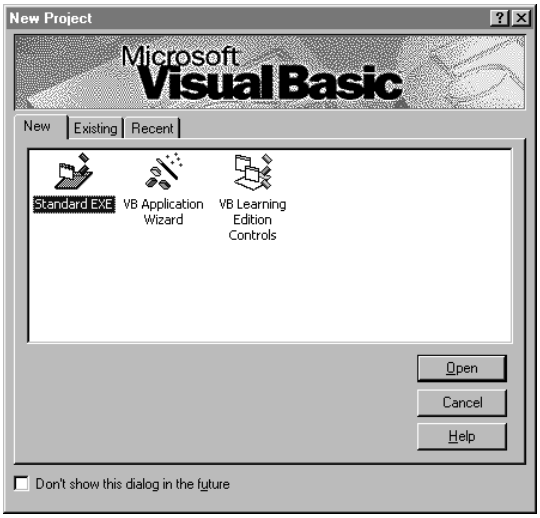
Visual Basic -ohjelmointiympäristö sisältää kaikki työvälineet, joita tarvitset tehdäksesi tehokkaita Windows-ohjelmia nopeasti ja vaivatta. Käynnistä Visual Basic nyt seuraavien ohjeiden mukaisesti.

HUOMAA Ellet ole vielä asentanut tämän kirjan mukana toimitettuja harjoitustiedostoja, noudata ohjeita, jotka ovat tämän kirjan alun kohdissa ”Sopivan aloituskohdan valitseminen” ja ”Harjoitustiedostojen asentaminen ja käyttö”. Palaa sitten tähän harjoitukseen.

Visual Basicin käynnistäminen

1. Napsauta Windows 95:n Käynnistä-painiketta, valitse Ohjelmat (Programs) ja Microsoft Visual Basic 5.0 -kansio. Kansion kuvakkeet näkyvät luettelossa.
2. Napsauta Visual Basic 5.0 -ohjelmakuvaketta.

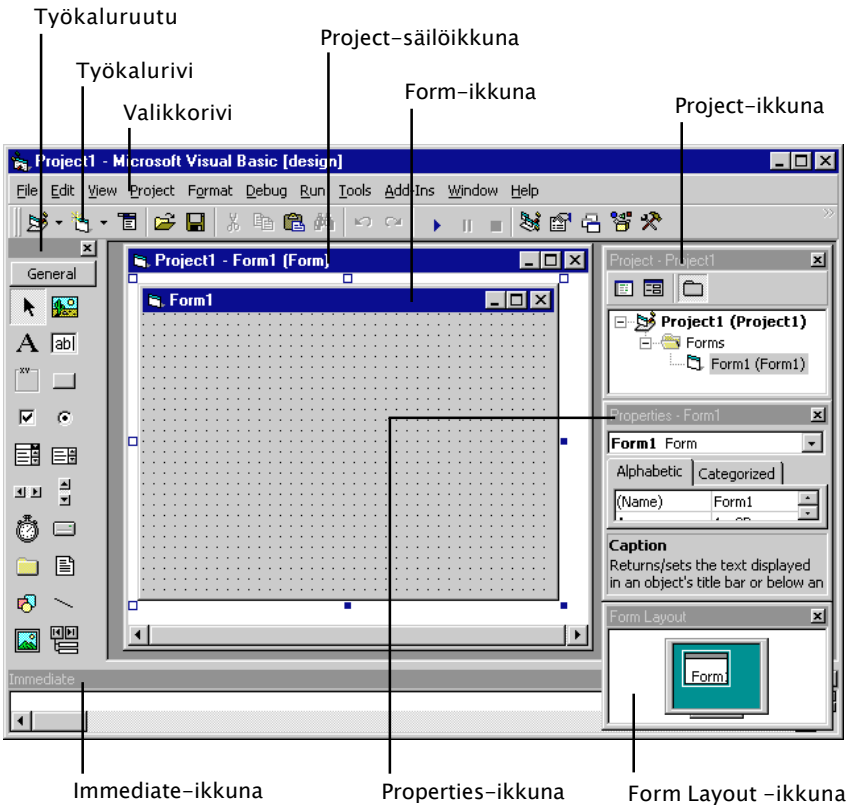
   New Project -valintaikkuna avautuu seuraavan kuvan mukaisena. Siinä pyydetään valitsemaan luotavan ohjelmointiprojektin tyyppi.



3. Hyväksy uusi oletusprojekti, joka on Standard EXE eli tavallinen 32-bittinen Visual Basic -ohjelma, napsauttamalla Open.

   Uusi projekti avautuu Visual Basic -ohjelmointiympäristöön. Siinä on ikkunoita ja työkaluja, jotka on esitetty viereisen sivun kuvassa.

Visual Basic -ohjelmointiympäristössä on ohjelmointityökaluja, jotka auttavat Visual Basic -ohjelmien kehittämisessä. *Valikkoriviltä* valitaan useimmat ohjelmointiympäristöön vaikuttavat komennot. Valikot ja komennot toimivat samalla tavoin kuin kaikissa muissakin Windows-ohjelmissa, ja voit valita niitä näppäimillä tai hiirellä. Valikkorivin alapuolella on *työkalurivi* eli



Jos haluat lukea työkalurivin painikkeen toiminnan kuvauksen, pidä hiiren osoitinta vähän aikaa painikkeen päällä.

joukko painikkeita, joilla valitaan nopeasti komentoja ja ohjataan Visual Basic -ohjelmointiympäristöä. Jos olet käyttänyt Microsoft Exceliä tai Microsoft Wordia, työkalurivin pitäisi olla sinulle tuttu. Kun haluat valita työkalurivin painikkeen, napsauta sitä hiirellä. Näytön alareunassa on Windows 95:n tehtäväpalkki. Sen avulla voit siirtyä Visual Basicin osasta toiseen ja käynnistää muita Windows-ohjelmia.

Käytettävissäsi on myös työkaluruutu, valikkorivi, Project-säilöikkuna, Form-ikkuna, Project-ikkuna, Immediate-ikkuna, Properties-ikkuna ja Form Layout -ikkuna. Näiden ikkunoiden tarkat koot ja muodot riippuvat järjestelmän asetuksista. Visual Basic 5:ssä voit kohdistaa ikkunoita ja kytkeä eli *telakoida* niitä toisiinsa, jotta ohjelmointijärjestelmän kaikki osat ovat näkyvissä ja helposti käsillä. Opit mukauttamaan ohjelmointiympäristöä myöhemmin tässä luvussa.

Seuraavaksi voit kokeilla valikkorivin ja työkalurivin käyttämistä ladatessasi muistiin ja suorittaessasi Visual Basic -esimerkkiohjelman nimeltä StepUp.

Ohjelmointiprojektin avaaminen käyttäen valikkoriviä

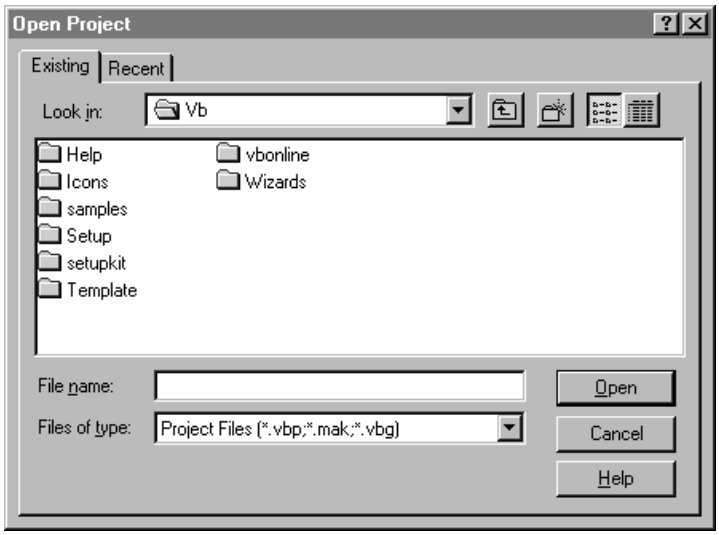
1 Napsauta File-valikkoa hiirellä.

File-valikon sisältö tulee näkyviin.

2 Valitse File-valikosta komento Open Project.

Open Project -valintaikkuna avautuu. Siitä voit avata minkä tahansa Visual Basic -ohjelman, joka on tallennettu kiintolevylle, verkkoasemaan, CD-tietolevylle tai levykkeelle:

Visual Basic -projektitiedostojen tarkenne on .vbp, .mak tai .vbg.



Ylempi taso -painike

3 Napsauta Ylempi taso -painiketta kolmesti, kaksoisnapsauta päähakemiston kansiota Vb5Sbs ja kaksoisnapsauta kansiota Less01.

Kansio \Vb5Sbs (oletuskansio, jonka tämän kirjan harjoitustiedostojen asennusohjelma luo) sisältää kirjan kaikki harjoitus- ja esimerkkitiedostot. Tarvitset kirjan kussakin luvussa kansiota Less*xx*, jossa *xx* on luvun numero.

4 Valitse StepUp-projekti ja napsauta Open.

Visual Basic lataa StepUp-projektitiedoston mukana StepUp-ohjelman käyttöliittymälomakkeen, ominaisuudet, ohjelmakäskyt ja vakiomoduulin.

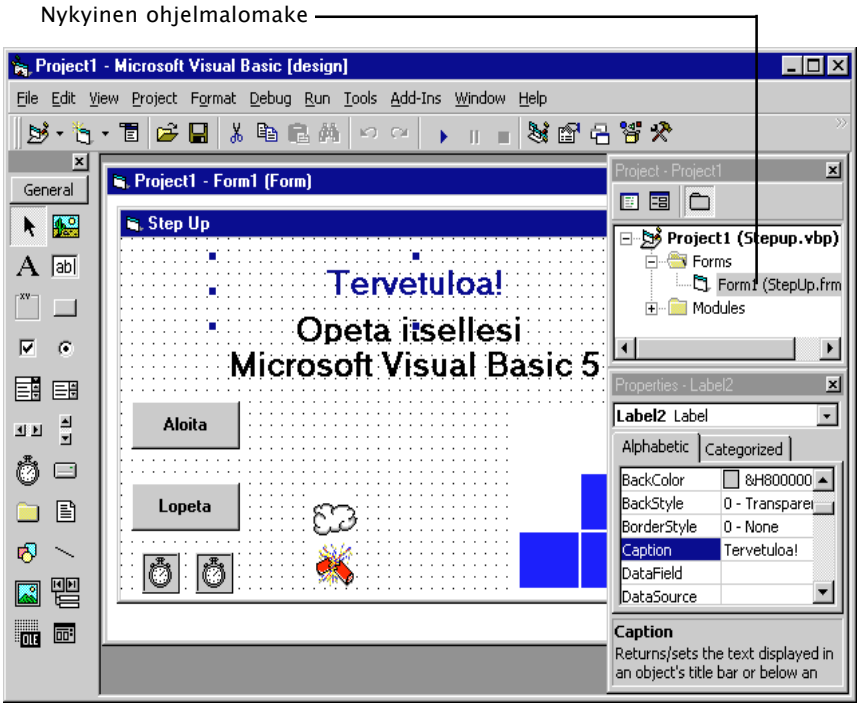
5 Ellei StepUp-lomaketta näy, kaksoisnapsauta Project-ikkunan Forms-kansiota ja sitten lomaketta Form1 (StepUp.frm).

Voidaksesi käsitellä projektin osaa sinun pitää valita se Project-ikkunasta.

Näytä objekti -painike

6 Napsauta Project-ikkunan Näytä objekti -painiketta, jotta voit tarkastella ohjelman käyttöliittymää.

Ohjelman lomake tulee näkyviin seuraavan kuvan kaltaisena:



Ellet näe Project-säilöikkunaa lomakkeen ympärillä, se on suurennettu, ja sinun pitää napsauttaa työkalurivin Palauta ikkuna -painiketta nähdäksesi projektin kuvan mukaisena. StepUp on hyvin yksinkertainen Visual Basic -ohjelma, jonka tarkoitus on selventää perusasioita. Koska StepUp-ohjelma sisältää useita tavanomaisen Visual Basic -ohjelman osia, voit käyttää sitä hyväksi tutustuessasi ohjelmointiympäristöön. Kun käynnistät ohjelman, se näyttää animaation ja toivottaa sinut tervetulleeksi tämän kirjan pariin.

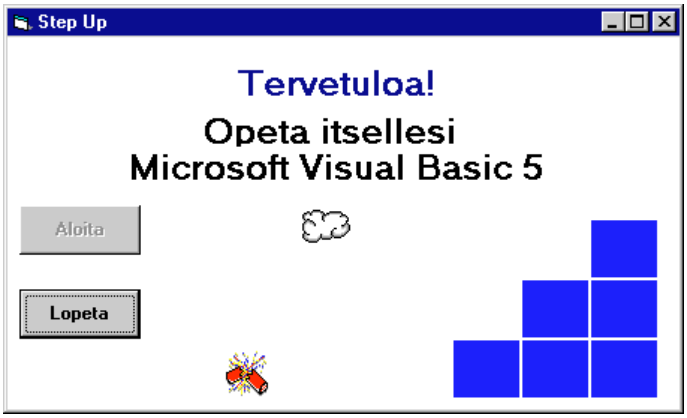
Käynnistä-painike

7 Napsauta Visual Basic -työkalurivin Käynnistä-painiketta, jotta StepUp-ohjelma käynnistyy Visual Basic -ympäristössä.

Työkaluruutu ja Properties-ikkuna katoavat, ja StepUp-ohjelma käynnistyy.

8 Napsauta Aloita-painiketta, jotta näet yksinkertaisen animaation. Näyttö muuttuu seuraavan sivun kuvan mukaiseksi.

9 Lopeta ohjelma ja palaa ohjelmointiympäristöön napsauttamalla Lopeta-painiketta. Siinä olikin kaikki — olet juuri suorittanut ensimmäisen Visual Basic -ohjelmasi!



Siirto-, telakointi- ja koonmuuttotyökalut

Kun näytössä on seitsemän ohjelmointityökalua, Visual Basic -ohjelmointiympäristö voi näyttää aika sekavalta. Visual Basic 5:ssä onkin mahdollista suurentaa, pienentää, siirtää ja telakoida ohjelmointityökaluja, jotta saat näytön osat muodoltaan ja kooltaan sopiviksi.

Voit käyttää Visual Basic 5:ssä uutta telakoinniksi kutsuttua toimintoa järjestäessäsi ohjelmointityökaluja.

Kun haluat siirtää ikkunaa, työkaluriviä tai työkaluruutua, napsauta sen otsikkoriviä ja vedä se uuteen paikkaan. Jos siirrät ikkunan toisen ikkunan reunaan, se kiinnittyy eli telakoituu tähän automaattisesti. Telakoitavat ikkunat ovat hyödyllisiä, koska ne pysyvät aina näkyvissä (eli eivät jää piiloon toisten ikkunoiden taakse).

Jos haluat nähdä suuremman osan telakoidusta ikkunasta, vedä sen reunaa ulospäin. Kun haluat katsella telakoitujen ikkunoiden sijasta limittäisiä ikkunoita samalla tavoin kuin Visual Basicin edellisissä versioissa, valitse Tools-valikosta Options, napsauta Docking-välilehteä ja poista valintamerkki jokaisesta työkalusta, jonka pitää toimia muista erillään.

Kun luet seuraavia kohtia, kokeile Visual Basic -ohjelmointiympäristön eri työkalujen siirtämistä, telakoimista ja koon muuttamista, kunnes nämä tehtävät sujuvat luontevasti.

Käyttöliittymälomake

Kukin lomake vastaa käyttöliittymän ikkunaa.

Visual Basicissa *lomake* on ikkuna, jota muokkaamalla muodostetaan ohjelman käyttöliittymä. StepUp-ohjelmassa lomake on ikkuna, jonka näet ohjelman suorituksen aikana. Lomakkeessa voi olla valikoita, painikkeita, luetteloruutuja, vierityspalkkeja ja kaikkia muita Windows-ohjelmista tuttuja osia. Kun käynnistät Visual Basic -ohjelmointiympäristön, näyttöön tulee oletuslomake Form1 ja tavallinen taustaruudukko (tasaisin välein sijoitettujen pisteiden joukko) ohjelman käyttöliittymän suunnittelemista ja asettelemista varten. Voit muuttaa lomakkeen kokoa hiirellä; lomake voi varata näytön alan kokonaan tai osittain. Voit muodostaa lisälomakkeita Project-valikon komennolla Add Form.

Jos osa lomakkeesta jää ohjelmointityökalujen taakse, voit poistaa työkalut näkyvistä tai pienentää niitä niin, että ne vievät vähemmän tilaa, tai napsauttaa lomakkeen otsikkoriviä ja vetää lomaketta, kunnes piilossa olleet osat tulevat esiin. Visual Basicin edellisistä versioista poiketen lomakkeen siirtäminen Visual Basic 5:n ikkunassa ei vaikuta lomakkeen sijaintiin ohjelman suorituksen aikana. Ikkunan ajonaikaisen paikan määrää Form Layout -ikkuna. Kun haluat osoittaa uuden lomakkeen paikan, vedä Form Layout -ikkunan pieni esikatselulomake sopivaan kohtaan.

Työkaluruutu

Voit lisätä ohjelman käyttöliittymän osia lomakkeeseen käyttämällä työkaluruudun työkaluja. Työkaluruutu avautuu napsauttamalla työkalurivin Työkaluruutu-painiketta. Se sijaitsee tavallisesti näytön vasemmassa reunassa. Työkalulla lisätään ohjelmaan *objekti* eli ohjelmoitava käyttöliittymän osa. Työkaluruudussa on työkaluja, joilla lisätään käyttöliittymään kuvia, tekstejä, painikkeita, luetteloruutuja, vierityspalkkeja, valikoita ja geometrisia kuvioita. Nämä osat näkyvät käyttäjälle ohjelman suorituksen aikana ja toimivat samalla tavoin kuin kaikissa muissakin Windows-ohjelmissa.

Työkaluruutu-painike

Voit siirtää työkaluruutua näytössä osoittamalla sen otsikkoriviä, pitämällä hiiren vasenta painiketta alhaalla ja vetämällä.

Työkaluruudun työkaluilla voidaan lisätä ohjelmaan myös sellaisia objekteja, jotka toimivat Visual Basic -ohjelman ”kulissien takana”. Nämä tehokkaat objektit suorittavat tärkeitä tehtäviä mutta eivät näy käyttäjälle ohjelman ollessa käynnissä; ne esimerkiksi käsittelevät tietokantoja, toimivat yhdessä muiden Windows-ohjelmien kanssa tai tarkkailevat ajan kulumista.

Voit tuoda työkaluruudussa olevan työkalun nimen esiin pitämällä hiiren osoitinta hetken työkalun päällä. Pääset kokeilemaan työkaluruudun toimintaa myöhemmin tässä harjoituksessa.

Properties-ikkuna

Properties-ikkunan avulla voit muuttaa käyttöliittymän osien *ominaisuusasetuksia* lomakkeessa. Ominaisuusasetus tarkoittaa käyttöliittymän osan laatua. Esimerkiksi StepUp-ohjelman tervetulotoivotuksen fonttia, fonttikokoa tai kohdistusta voidaan muuttaa. (Voit esittää Visual Basicissa tekstin millä tahansa järjestelmään asennetulla fontilla samalla tavoin kuin Excelissä tai Wordissa.) Voit muuttaa ominaisuusasetuksia käyttämällä Properties-ikkunaa käyttöliittymää suunnitellessasi tai ohjelmallisesti ohjelman suorituksen aikana.

Properties-ikkunassa on avattava objektiluettelo, jossa on mainittu lomakkeen kaikki käyttöliittymän osat (objektit). Ikkunassa on lueteltu myös jokaisen objektin muokattavissa olevat ominaisuusasetukset. (Luettelo voi olla aakkosten tai luokkien mukaisessa järjestyksessä.) Kokeile nyt StepUp-ohjelman Lopeta-painikkeen Caption-ominaisuuden muuttamista.

Ominaisuuden muuttaminen

1 Tarkista, että StepUp-ohjelman suoritus on päättynyt (otsikkorivillä on sana *design*, kun ohjelma ei ole enää käynnissä) ja napsauta lomakkeen Lopeta-objektia.

Lopeta-objekti (komentopainike) on ympäröity suorakulmiolla eli *valittu*. Jotta voisit muokata Visual Basic -lomakkeen objektia, sinun on valittava se ensin.

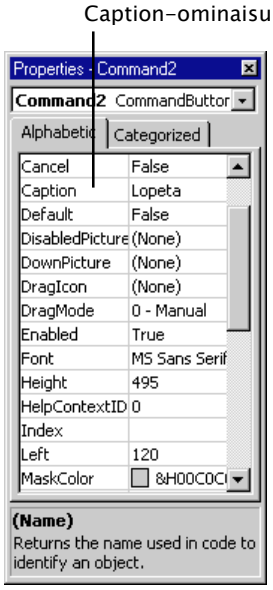
Napsauta työkalurivin Properties-ikkuna-painiketta.

Ohjelmointiympäristön Properties-ikkuna muuttuu korostetuksi. (Ellei se ollut näkyvissä, se avautuu nyt.)

Properties-ikkuna-painike

2 Kaksoisnapsauta Properties-ikkunan otsikkoriviä, jotta ikkuna muuttuu kelluvaksi (telakoimattomaksi).

Näet seuraavan kuvan kaltaisen ikkunan:



Properties-ikkunassa on lueteltu lomakkeen toisen komentopainikkeen kaikki ominaisuusasetukset. (Komentopainikkeelle on valittavissa kaikkiaan 32 ominaisuutta.) Ominaisuuksien nimet on lueteltu ikkunan vasemmassa sarakkeessa ja nykyiset arvot oikeassa sarakkeessa. Alphabetic-välilehti sisältää ominaisuudet aakkosjärjestyksessä.

3 Vieritä luetteloruutua, kunnes näet Caption-ominaisuuden.

Properties-ikkunaa vieritetään samoin kuin tavallista luetteloruutua.

4 Kaksoisnapsauta Caption-ominaisuutta (näkyy vasemmassa sarakkeessa).

Painikkeen nykyinen teksti (”Lopeta”) näkyy korostettuna oikeassa sarakkeessa, ja kohdistin vilkkuu sen oikealla puolella.

5 Paina DEL, kirjoita **Poistu** ja paina ENTER.

Caption-ominaisuuden arvo on tämän jälkeen ”Poistu” eikä ”Lopeta . Lomakkeessa näkyvä painikkeen teksti vaihtuu, ja kun seuraavan kerran käynnistät ohjelman, painikkeessa lukee *Poistu*.

6 Palauta Properties-ikkuna Project-ikkunan yläpuolelle ja telakoi se.

Nyt on sopiva hetki harjoitella ikkunoiden telakoimista. Otsikkorivin kaksoisnapsauttaminen on nopein telakoimistapa, ja voit hienosäätää ikkunan paikkaa vetämällä sen Project-ikkunan reunan yläpuolelle. Koska näytössä on niin monia ikkunoita lähellä toisiaan, ”manuaalinen telakointi vaatii hieman harjoittelua ja voi aluksi tuntua hankalalta. Kun myöhemmin käytät Visual Basicin ohjelmointityökaluja, sinulle on paljon hyötyä hyvin järjestetystä työtilasta, ja opettelemiseen kuluttamasi aika osoittautuu hyvinkin vaivan arvoiseksi.

Ominaisuuksista puhuen

Visual Basicissa ohjelman käyttöliittymän jokaisella osalla (myös itse lomakkeella) on joukko muutettavia ominaisuuksia. Voit määrittää ominaisuudet ohjelmaa suunnitellessasi Properties-ikkunassa ja viitata niihin ohjelmalauseilla ohjelmaa suoritettaessa. (Käyttöliittymän osat, jotka vastaanottavat syöttötietoja, välittävät usein tiedot edelleen ominaisuuksien kautta.) Aluksi ominaisuudet voivat kuitenkin tuntua vaikeasti ymmärrettävältä käsitteeltä. Niiden vertaaminen arkipäivän esineisiin on ehkä avuksi.

Käytetään vertailukohtana polkupyörää: polkupyörä on esine, jolla siirrytään paikasta toiseen. Koska se on fyysinen objekti, sillä on useita olennaisia piirteitä. Sillä on mallinimi, väri, vaihteet, jarrut ja pyörät, ja se kuuluu tiettyyn luokkaan (se on esimerkiksi retkipyörä, maastopyörä tai tandem). Visual Basicin termejä käyttäen nämä ovat polkupyöräobjektin *ominaisuuksia*. Mallinetta, jonka avulla polkupyörän runko on koottu, voitaisiin kutsua polkupyörätyökaluksi. Useimmat polkupyörän ominaisuudet määritetään kokoamisvaiheessa, kun taas eräät (kuten renkaat, nopeus ja ikä) muuttuvat polkupyörää käytettäessä. Visual Basicissa joudut tekemisiin kummankinlaisten ominaisuuksien kanssa.

Project-ikkuna

Visual Basic -ohjelma muodostuu useasta tiedostosta, jotka kootaan yhteen eli *käännetään*, kun ohjelma on valmis. Jotta voisit helposti siirtyä tiedostosta toiseen projektia muokatessasi, Visual Basicin suunnittelijat ovat lisänneet ohjelmointiympäristöön *Project-ikkunan*. Siinä on lueteltu kaikki ohjelmointiprosessiin liittyvät tiedostot ja kaksi painiketta, joilla niitä voidaan valita: Näytä lauseet ja Näytä objekti. Voit lisätä, poistaa ja tallentaa projektin tiedostoja yksitellen käyttämällä File- ja Project-valikoiden komentoja. Projektiin tehdyt muutokset näkyvät Project-ikkunassa.

Tiedostoa, joka sisältää kaikkien ohjelmointiprojektissa tarvittavien tiedostojen luettelon, sanotaan *Visual Basic -projektitiedostoksi (.vbp)*. Visual Basic 5:ssä on mahdollista ladata Project-ikkunaan useita projektitiedostoja samanaikaisesti ja siirtyä projektitiedostosta toiseen napsauttamalla projektin nimeä. Project-ikkunassa näkyy kunkin projektin nimen alla projektin osat puumaisena rakenteena, joka muistuttaa Windows Resurssienhallinnan (Explorer) jäsennysnäkymää. Voit laajentaa ja supistaa puun haaroja, myös Forms-, Modules- ja muita luokkia, napsauttamalla kansioiden vieressä olevia plus- ja miinusmerkkejä.

Seuraavassa harjoituksessa tutustut StepUp-ohjelman Project-ikkunaan.

Project-ikkunan näyttäminen

Project-ikkuna-painike

1. Napsauta työkalurivin Project-ikkuna-painiketta.

   Ohjelmointiympäristön Project-ikkuna eli Project Explorer muuttuu korostetuksi. (Ellei ikkuna ollut näkyvissä, se avautuu nyt.)

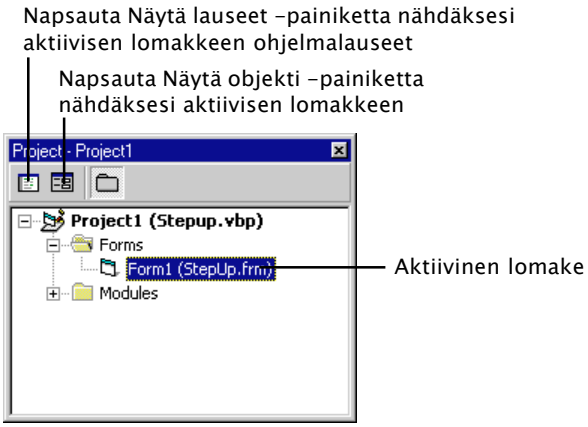
2. Kaksoisnapsauta Project-ikkunan otsikkoriviä, jotta ikkuna muuttuu kelluvaksi (telakoimattomaksi).

   Näet Project-ikkunan, joka muistuttaa viereisellä sivulla esitettyä.

3. Napsauta Forms- ja Modules-kansioiden vieressä olevia plusmerkkejä (ellet ole vielä napsauttanut niitä), jotta näet projektin kaikki osat.

   Tämän ohjelmointiprojektin projektitiedosto on nimeltään StepUp. StepUp-projektin kohdalla on mainittu tiedostot StepUp.frm ja StepUp.bas. StepUp.frm sisältää käyttöliittymälomakkeen ja tämän objekteihin liittyvät ohjelmalauseet. StepUp.bas sisältää ohjelmalauseet, jotka ovat yhteisiä ohjelman kaikille osille. Kun ohjelma käännetään suoritettavaksi tiedostoksi tai valmistellaan Windowsissa suorittamista varten, nämä tiedostot kootaan yhdeksi .exe-tiedostoksi.

4. Kaksoisnapsauta Project-ikkunan otsikkoriviä, jotta se muuttuu jälleen telakoiduksi.



Ohjeiden etsiminen

Visual Basicissa on elektroninen käyttöopas, josta löydät lisätietoja Visual Basicin ohjelmointiympäristöstä, kehitystyökaluista ja ohjelmointikielestä. Tutustu Visual Basicin ohjejärjestelmään hieman ennen kuin kirjoitat ensimmäisen ohjelmasi.

Voit tuoda näyttöön ohjeita usealla tavalla.

Kun tarvitset ohjeita	Toimi näin
Aiheen tai tehtävän mukaan	Valitse Visual Basicin Help-valikosta Microsoft Visual Basic Help Topics.
Käyttäessäsi ikkunaa tai valintaikkunaa	Paina F1 tai napsauta valintaikkunan Help-painiketta.
Tietystä ohjelmointityökalusta, ominaisuudesta tai ohjelmointikielen osasta.	Valitse Help-valikosta Microsoft Visual Basic Help Topics, valitse Index-välilehti ja kirjoita etsittävä sana tekstiruutuun.
Yhteyden ottamisesta Microsoftin tuotetukeen	Valitse Help-valikosta Obtaining Technical Support. Jos sinulla on Internet-yhteys, voit myös avata Microsoft On The Web -alivalikon ja ottaa yhteyden johonkin luetelluista Visual Basicin Web-paikoista.

Noudata seuraavia ohjeita, kun tarvitset ohjeita tietystä Visual Basicia koskevasta aiheesta. Tässä kerrotaan, miten löydät Project-ikkunaan liittyviä tietoja, mutta voit valita minkä tahansa haluamasi aiheen.

Tiettyä aihetta koskevan ohjeen etsiminen

Help-valikko on portti Visual Basicin ohje-järjestelmään.

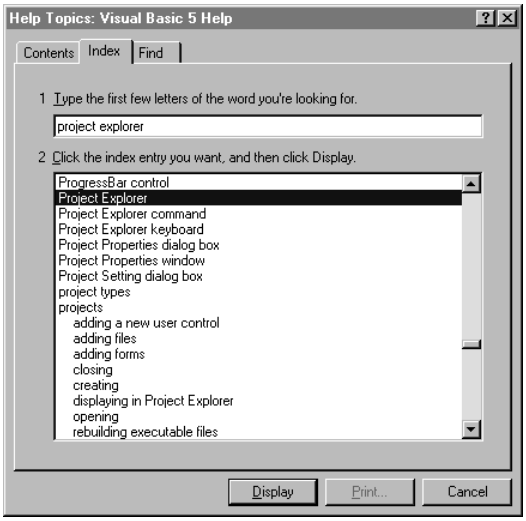
1. Napsauta valikkorivin Help-valikkoa.

   Help-valikon sisältö tulee näkyviin.

2. Valitse Help-valikosta komento Microsoft Visual Basic Help Topics.

3. Varmista, että Index-välilehti on valittu, ja kirjoita tekstiruutuun **project explorer** (tai jokin muu hakusana).

   Kun kirjoitat sanat *project explorer,* luetteloruutuun tulevat aiheet, jotka avautuvat kirjaimella ”p”, kirjaimilla ”pr” ja niin edelleen, kunnes valintaikkuna on seuraavan kuvan mukainen:



4. Kaksoisnapsauta aihetta Project Explorer.

   Help-ikkunaan tulee tietoja Visual Basicin Project-ikkunasta. Elleivät ne kaikki mahdu näkyviin, ikkunan reunoissa on vierityspalkit, joita napsauttamalla saat esille tekstin loppuosan.

5. Napsauta tekstiä See Also, joka on ikkunan yläosassa ja alleviivattu vihreällä.

   Topics Found -ponnahdusikkuna avautuu. Kaksoisnapsauta mitä tahansa Project-ikkunaan liittyvää aihetta, josta olet kiinnostunut.

6. Sulje Topics Found -ponnahdusikkuna painamalla ESC.

7. Poistu ohjejärjestelmästä napsauttamalla Help-ikkunan otsikkorivin sulkemispainiketta.

Visual Basicin ohjejärjestelmästä on hyötyä, kun haluat tietoja ohjelmointiympäristöstä tai mistä tahansa Visual Basic -ohjelmointiin kuuluvasta aiheesta. Muista kokeilla sitä, jos sinulla on kysyttävää.

Lucky Seven: ensimmäinen Visual Basic -ohjelmasi

Nyt kun olet hieman tutustunut Visual Basic -ohjelmointiympäristöön, voit ryhtyä tekemään ensimmäistä Visual Basic -ohjelmaasi. Tarkoitus on kirjoittaa peliohjelma Lucky Seven, joka jäljittelee peliautomaattia. Ohjelma perustuu yksinkertaiseen käyttöliittymään ja voidaan kirjoittaa ja kääntää Visual Basicilla muutamassa minuutissa. (Jos haluat kokeilla ohjelmaa Lucky.exe ennen kuin aloitat, löydät sen kiintolevyn kansiosta \Vb5Sbs\Less01.) Valmis ohjelma näyttää tällaiselta:

HUOMAA Jos haluat lisätietoja siitä, mitä ohjelmat ovat ja miten Visual Basic -ohjelmia tehdään, katso tämän kirjan liitettä ”Ajatteleminen ohjelmoijan tavalla”.

Ohjelmoimisen vaiheet

Lucky Seven -ohjelman käyttöliittymään kuuluu kaksi komentopainiketta, kolme numerokehystä, kolikkopinon kuva ja teksti Lucky Seven. Nämä osat muodostettiin luomalla Lucky Seven -lomakkeeseen seitsemän objektia ja muuttamalla niiden useita ominaisuuksia. Kun käyttöliittymä oli saatu valmiiksi, Spin- ja End-komentopainikkeisiin lisättiin ohjelmalauseet painikkeiden napsautusten ja satunnaislukujen tuottamista varten. Kun toistat nämä tehtävät luodessasi Lucky Seven -ohjelmaa, suoritat kolme vaihetta, joita kuvataan liitteessä ”Ajatteleminen

ohjelmoijan tavalla": käyttöliittymän muodostaminen, ominaisuuksien määrittäminen ja ohjelmalauseiden kirjoittaminen. Lucky Seven -ohjelmaan liittyviä vaiheita kuvataan seuraavassa taulukossa:

Ohjelmoinnin vaihe	Kohteiden määrä
1. Muodosta käyttöliittymä.	7 objektia
2. Määritä ominaisuudet.	10 ominaisuutta
3. Kirjoita ohjelmalauseet.	2 objektia

Toinen tapa hahmottaa Lucky Seven -ohjelma on käyttää seuraavaa algoritmia eli ohjelman toiminnan kuvausta. Algoritmin laatiminen on usein hyödyllinen lähtökohta ohjelman kehittämiselle.

Käyttöliittymän muodostaminen

Aloita Lucky Seven -ohjelman kirjoittaminen luomalla projekti ja muodostamalla käyttöliittymä työkaluruudussa olevista objekteista.

Käyttöliittymän muodostaminen

Aloita uusi projekti valitsemalla File-valikosta komento New Project.

1. Valitse File-valikosta komento New Project.

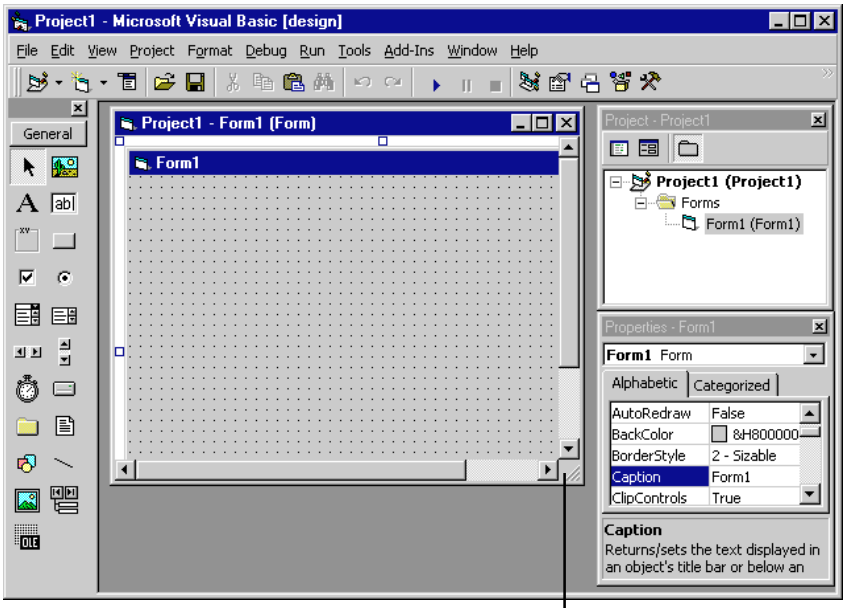
   Vastaa No, jos Visual Basic kysyy, haluatko tallentaa StepUp-ohjelmaan tekemäsi muutokset. StepUp-ohjelma katoaa muistista.

2. Luo tavallinen 32-bittinen Visual Basic -sovellus napsauttamalla OK.

   Visual Basic luo tyhjän ohjelmointiprojektin ja tuo näytön keskelle tyhjän lomakkeen, johon voit muodostaa ohjelman käyttöliittymän. Suurenna lomaketta ja lisää siihen käyttöliittymän kaksi painiketta.

3. Siirrä hiiren osoitin Form-ikkunan (ei Project-säilöikkunan) oikeaan alakulmaan, jotta se muuttuu koonmuutto-osoittimeksi, ja suurenna lomaketta, jotta siinä on tilaa ohjelman objekteille.

   Kun muutat lomakkeen kokoa, Project-ikkunaan ilmestyy vierityspalkit, kuten seuraavasta kuvasta näkyy.



Muuta lomakkeen kokoa vetämällä

Tuo koko lomake näkyviin muuttamalla Project-säilöikkunan kokoa niin, että vierityspalkit katoavat, ja sulkemalla tai siirtämällä Properties-, Project- ja Form Layout -ikkunaa. Lisää lomakkeeseen komentopainikeobjekti.

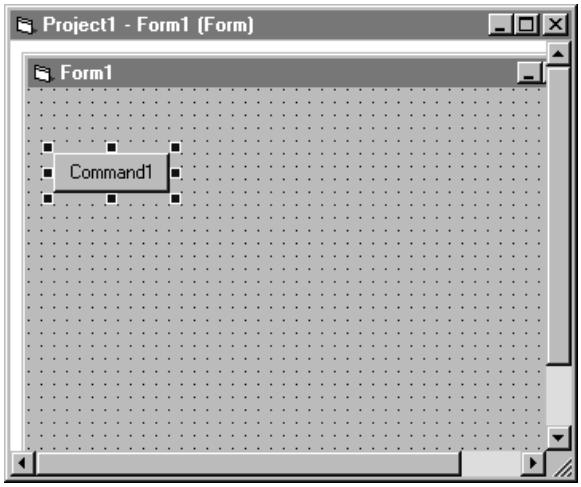
Komentopainike-työkalu

4 Napsauta työkaluruudun Komentopainike-työkalua ja siirrä hiiren osoitin lomakkeeseen.

Komentopainike-työkalu on valittu, ja hiiriosoitin muuttuu pieneksi ristiksi ollessaan lomakkeen sisäpuolella. Osoittimen muodon tarkoitus on helpottaa komentopainikkeen suorakulmaisen muodon piirtämistä. Kun pidät hiiren vasenta painiketta alhaalla ja vedät hiirtä, Visual Basic piirtää komentopainikkeen ja kohdistaa sen yhteen lomakkeen taustaruudukon pisteistä. Yritä piirtää ensimmäinen komentopainike nyt.

5 Siirrä hiiren osoitin lähelle lomakkeen vasenta yläkulmaa, pidä hiiren vasenta painiketta alhaalla ja vedä hiirtä alas ja oikealle. Lopeta vetäminen ja vapauta hiiren painike, kun lomakkeessa näkyy seuraavan kuvan kaltainen komentopainike:

Komentopainikkeen nimi on Command1.



Lomakkeeseen ilmestyy komentopainike, jonka ympärillä on valintakahvat. Sen nimi on Command1, joka merkitsee ohjelman ensimmäistä komentopainiketta. (Sinun kannattaa painaa nimi mieleesi — näet sen uudelleen myöhemmin, kun kirjoitat ohjelmalauseita.)

Voit siirtää komentopainikkeita hiirellä ja muuttaa niiden kokoa vetämällä valintakahvoja, kun Visual Basic on *suunnitteluti lassa* (Visual Basic -ohjelmointiympäristö on aktiivinen). Kun ohjelma on käynnissä, käyttäjä ei voi siirtää käyttöliittymän osia, ellet ole määrittänyt ohjelmassa erityisominaisuutta, joka sallii tämän. Kokeile komentopainikkeen paikan ja koon muuttamista nyt.

Komentopainikkeen paikan ja koon muuttaminen

Taustaruudukko helpottaa käyttöliittymän suunnittelemista.

1. Vedä komentopainiketta oikealle hiirellä.

   Komentopainike sijoittuu lähimpään taustaruudukon pisteeseen, kun vapautat hiiren painikkeen. Taustaruudukon tehtävä on helpottaa käyttöliittymän osien muokkaamista ja kohdistamista. Jos haluat muuttaa sen tiheyttä, valitse Tools-valikosta komento Options ja napsauta General-välilehteä.

2. Siirrä hiiren osoitin komentopainikkeen oikeaan alakulmaan.

   Kun osoitin on valitun objektin kulmassa tai reunaviivalla, se muuttuu koonmuutto-osoittimeksi. Sillä muutetaan objektin mittoja.

3. Suurenna komentopainiketta pitämällä hiiren vasenta painiketta alhaalla ja vetämällä osoitinta alas ja oikealle.

   Kun vapautat hiiren painikkeen, komentopainikkeen koko muuttuu, ja sen kulma sijoittuu lähimpään taustaruudukon pisteeseen.

4. Palauta komentopainike alkuperäiseen kokoonsa käyttäen koonmuutto-osoitinta ja siirrä painike sitten alkuperäiseen paikkaansa lomakkeessa.

Lisää lomakkeeseen toinen komentopainike ensimmäisen alapuolelle.

Toisen komentopainikkeen lisääminen

Komentopainike-työkalu

1. Napsauta työkalupaletin Komentopainike-työkalua.
2. Piirrä ensimmäisen komentopainikkeen alapuolelle toinen vetämällä hiirtä. (Tee painikkeista yhdenmukaisuuden vuoksi samankokoiset.)
3. Muuta komentopainikkeen kokoa tai paikkaa tarvittaessa. Jos teit virheen, poista komentopainike ja aloita uudelleen alusta.

Voit poistaa objektin valitsemalla sen lomakkeesta ja painamalla DEL-näppäintä.

Tekstikehysten lisääminen

Lisää tekstikehykset, joita ohjelma käyttää näyttäessään pelin numerot. Kun käyttäjä napsauttaa Pyöritä-painiketta, tekstikehysten sisälle tulee kolme satunnaista numeroa. Jos yksi niistä on seitsemän, käyttäjä voittaa pelissä.

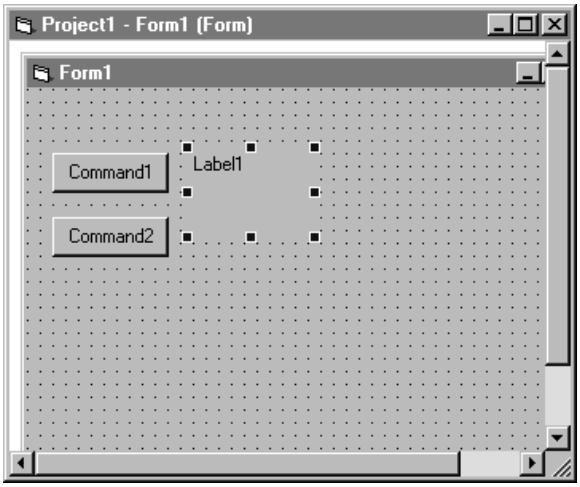
Teksti-työkalu

1. Napsauta työkaluruudun Teksti-työkalua ja siirrä hiiren osoite lomakkeeseen.

   Teksti-työkalu on valittu, ja osoitin muuttuu pieneksi ristiksi lomakkeen päällä.

2. Piirrä pieni nelikulmio, joka muistuttaa seuraavalla sivulla esitettyä.

   Luomasi tekstiobjektin nimi on Label1, koska se on ohjelman ensimmäinen tämäntyyppinen objekti. Tekstiobjekti on käyttöliittymän erityinen osa, jolla näytetään tekstiä, lukuja tai symboleita ohjelman toimiessa. Piirrä lomakkeeseen kaksi muuta tekstiobjektia, Label2 ja Label3.



HUOMAA Kun piirrät tekstiobjekteja, katso ponnahdusikkunaa, joka avautuu lisäämäsi tekstiobjektin viereen. Tätä ikkunaa, joka sisältää vaaka- ja pystymittoja, sanotaan koonmääritysikkunaksi. Ensimmäinen numero ilmaisee valitun objektin vasemman yläkulman sijainnin ja toinen numero objektin mitat. Numerot on esitetty käyttäen mittayksikköä nimeltä twip; yksi twip on typografisen pisteen kahdeskymmenesosa (piste on 1/72 tuumaa, joten twip on 1/1440 tuumaa eli noin 0,0176 mm). Koonmääritysikkunan avulla voit sijoittaa objekteja lomakkeeseen sekä verrata piirtämiesi objektien kokoja.

3. Napsauta Teksti-työkalua ja piirrä ensimmäisen tekstikehyksen oikealle puolelle toinen.

   Tee toisesta tekstikehyksestä samankokoinen kuin ensimmäinen. Siinä näkyy nimi ”Label2”.

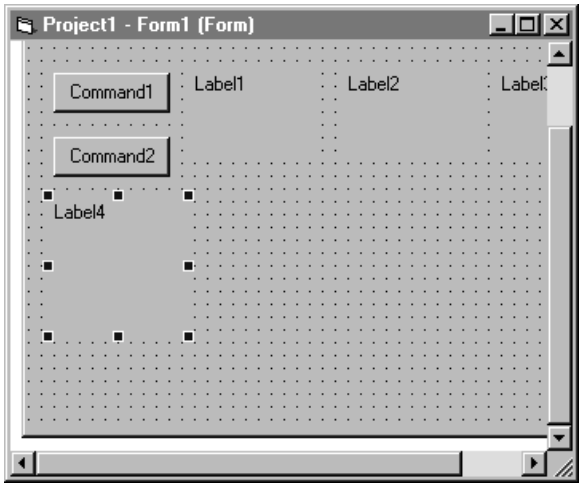
4. Napsauta Teksti-työkalua uudelleen ja piirrä toisen tekstikehyksen oikealle puolelle kolmas.

   Kehyksessä näkyy nimi ”Label3”. Lisää lomakkeeseen vielä neljäs ja viimeinen tekstikehys, johon sijoitetaan ohjelman nimi.

5. Napsauta Teksti-työkalua.

6. Piirrä komentopainikkeiden alapuolelle suuri suorakulmio.

Kun olet lopettanut, neljän tekstikehyksen pitäisi muistuttaa seuraavan sivun kuvaa. Voit muuttaa niiden kokoa, jos lomake ei näytä aivan oikealta.



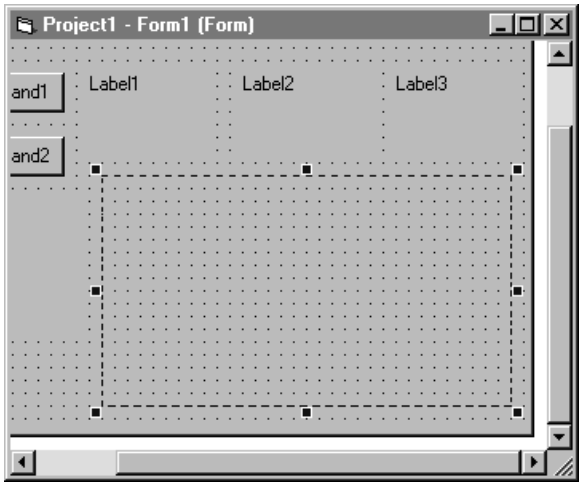
Lisää lomakkeeseen kuvakehys, johon tulee kolikkopinon kuva, jos ainakin yksi pelin numeroista on seitsemän eli pelaaja voittaa. Kuvakehyksen avulla näytetään ohjelmassa bittikarttoja, kuvakkeita ja muita kuvia. Yksi kuvakehyksen parhaista käyttökohteista on näyttää Visual Basiciin kuuluva leikekuva.

Kuvan lisääminen

Kuva-työkalu

1. Napsauta työkaluruudun Kuva-työkalua.
2. Piirrä tekstikehysten alapuolelle suuri suorakulmio.

Kun olet valmis, kuvakehyksen pitäisi olla seuraavan kuvan mukainen:



Tämän objektin nimi on Image1. Tarvitset tätä nimeä myöhemmin ohjelman lauseissa.

Nyt voit muuttaa käyttöliittymää määrittämällä ominaisuuksia.

Ominaisuuksien määrittäminen

Kuten aiemmin huomasit StepUp-ohjelmassa, voit muuttaa objektin ominaisuuksia valitsemalla objektin lomakkeesta ja muuttamalla sen asetuksia Properties-ikunassa. Aloita käyttöliittymän muokkaaminen muuttamalla kahden komentopainikkeen tekstit.

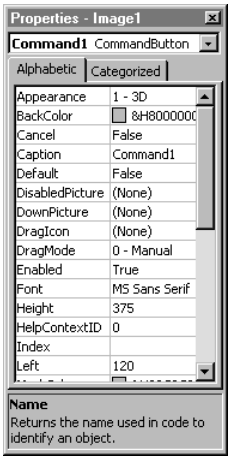
Komentopainikkeiden ominaisuuksien määrittäminen

1. Napsauta lomakkeen ensimmäistä komentopainiketta (Command1).

   Komentopainikkeen ympärille ilmestyy valintakahvat.

2. Kaksoisnapsauta Properties-ikkunan otsikkoriviä.

   Properties-ikkuna avautuu seuraavan kuvan näköisenä:



   Properties-ikkunassa on lueteltu ensimmäisen komentopainikkeen ominaisuusasetukset. Näihin kuuluvat esimerkiksi taustan väri, teksti, fontin korkeus ja komentopainikkeen leveys.

3. Kaksoisnapsauta Properties-ikkunan vasemmassa sarakkeessa olevaa Caption-ominaisuutta.

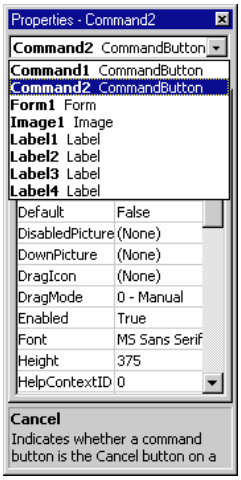
   Komentopainikkeen nykyinen teksti (”Command1”) muuttuu korostetuksi Properties-ikkunassa.

4. Kirjoita teksti **Pyöritä** ja paina ENTER.

   Tekstiksi tulee ”Pyöritä” sekä Properties-ikkunassa että lomakkeessa. Vaihda toisen komentopainikkeen tekstiksi ”Lopeta”. (Valitse painike tällä kertaa uudella tavalla.)

5 Avaa Properties-ikkunan yläosassa näkyvä objektiluettelo.

Luetteloon ilmestyy ohjelman käyttöliittymän objektien nimet:



6 Napsauta luettelon vaihtoehtoa Command2 (toinen komentopainike).

Toisen komentopainikkeen ominaisuusasetukset ilmestyvät Properties-ikkunaan.

7 Kaksoisnapsauta nykyistä teksti- eli Caption-ominaisuutta (”Command2), kirjoita **Lopeta** ja paina ENTER.

Toisen komentopainikkeen tekstiksi tulee ”Lopeta”.

HUOMAA Objektiluettelo on kätevä tapa siirtyä ohjelman objektista toiseen. Voit selata objekteja myös valitsemalla niitä lomakkeesta yksi kerrallaan napsauttamalla.

Määritä seuraavaksi ohjelman tekstiobjektien ominaisuudet. Kolmeen ensimmäiseen tekstiobjektiin sijoitetaan ohjelman tuottamat satunnaisluvut, ja niiden ominaisuusasetukset ovat yhtäläiset (voit määrittää useimmat niistä ryhmänä). Ohjelman nimen sisältävässä tekstiobjektissa tarvitaan hieman erilaiset ominaisuudet.

Kun haluat valita lomakkeesta useita objekteja, pidä VAIHTONÄPPÄINTÄ alhaalla ja napsauta objekteja.

Numerokehysten ominaisuuksien määrittäminen

1 Napsauta ensimmäistä numerolle tarkoitettua tekstiobjektia, pidä VAIHTONÄPPÄINTÄ alhaalla ja napsauta toista ja kolmatta tekstiobjektia. (Jos Properties-ikkuna on tiellä, siirrä se sivuun.)

Napsauttamiesi tekstiobjektien ympärille tulee valintakehys. Kun olet valinnut kaikki kolme tekstiobjektia, vapauta VAIHTONÄPPÄIN.

HUOMAA Kun useita objekteja on valittu, Properties-ikkunassa näkyvät vain ne ominaisuudet, jotka on mahdollista määrittää ryhmänä. Muuta tekstiobjektien Alignment-, BorderStyle- ja Font-ominaisuuksia, jotta tekstikehyksissä esitettävät numerot ovat keskitettyjä ja ympäröidyt suorakulmioilla ja jotta niiden fontti ja pistekoko ovat oikeat.

2 Napsauta Alignment- eli tasausominaisuutta ja sitten oikealla näkyvän avattavan luetteloruudun nuolipainiketta.

Luetteloruudussa näkyy joukko tasausasetuksia.

3 Valitse vaihtoehto 2 - Center eli keskitys.

Jokaisen valitun tekstiobjektin Alignment-ominaisuudeksi muuttuu 2 - Center. Muuta sitten BorderStyle-ominaisuutta.

4 Napsauta BorderStyle- eli reunaviivaominaisuutta ja sitten oikealla näkyvän avattavan luetteloruudun nuolipainiketta.

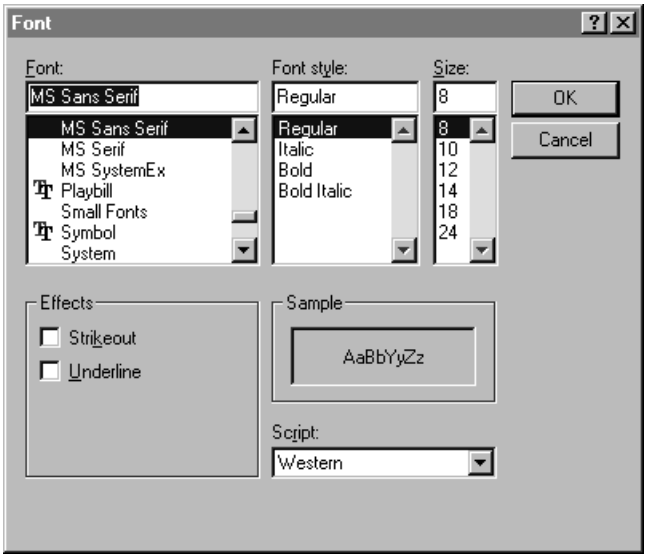
Luetteloruudussa näkyy kaksi reunaviiva-asetusta: 0 - None eli ei viivaa ja 1 - Fixed Single eli yksi viiva.

5 Lisää jokaisen valitun tekstiobjektin ympärille kapea reunaviiva valitsemalla vaihtoehto Click 1 - Fixed Single.

Vaihda tekstiobjektien fontti muuttamalla Font-ominaisuutta.

6 Kaksoisnapsauta Properties-ikkunassa näkyvää Font- eli fonttiominaisuutta.

Font-valintaikkuna avautuu tällaisena:



7. Vaihda fontiksi Times New Roman, tyyliksi Bold eli lihavoitu ja pistekooksi 24 ja napsauta OK.

   Tekstikehysten otsikot näkyvät nyt valitsemallasi fontilla, tyylillä ja koolla kirjoitettuina. Poista otsikot, jotta kehykset ovat tyhjiä ohjelman käynnistyessä. (Valitsemasi ominaisuusasetukset pysyvät voimassa, koska ne on tallennettu erikseen.) Tätä tehtävää varten sinun on valittava jokainen tekstiobjekti erikseen.

8. Poista valintakahvat kolmen tekstiobjektin ympäriltä napsauttamalla lomaketta ja napsauta ensimmäistä tekstiobjektia.

9. Kaksoisnapsauta Caption- eli otsikko-ominaisuutta ja paina DEL.

   Label1-objektin otsikko katoaa. Myöhemmin tässä harjoituksessa lisäät sen tilalle samannäköisen numeron käyttäen ohjelmalauseita.

10. Poista otsikot lomakkeen toisesta ja kolmannesta tekstiobjektista.

Kolme ensimmäistä tekstiobjektia ovat nyt valmiit. Vaihda seuraavaksi viimeisen tekstiobjektin Caption-, Font- ja ForeColor-ominaisuudet.

Otsikon ominaisuuksien määrittäminen

1. Napsauta lomakkeen neljättä tekstiobjektia.
2. Vaihda Caption- eli otsikko-ominaisuudeksi ”Onnen seiska”.
3. Kaksoisnapsauta Font-ominaisuutta ja vaihda Font-valintaikkunassa fontiksi Arial, fonttityyliksi Bold eli lihavoitu ja pistekooksi 20. Napsauta OK.

   Tekstiobjektin fontti vaihtuu. Huomaa, että objektin teksti jakautuu kahdelle riville, koska se ei enää mahdu yhdelle. Tämä on tärkeää: objektin sisällön on mahduttava objektin rajojen sisäpuolelle. Jos sisältö on liian suuri, se jakautuu eri riveille tai osa siitä rajautuu pois. Vaihda seuraavaksi tekstin edustaväri.

4. Kaksoisnapsauta Properties-ikkunan ForeColor-ominaisuutta.

   Luetteloruutuun ilmestyy System- ja Palette-välilehti, jotka ovat kaksi tapaa muuttaa objektin väriä. System-välilehti luettelee käyttöliittymän osien nykyiset värit. Luettelo vastaa työpöydän ominaisuusikkunan Ulkoasu-välilehden (Appearance) nykyisiä asetuksia. Palette-välilehti sisältää järjestelmän kaikki mahdolliset värit.

5. Napsauta Palette-välilehteä ja tumman purppuran väristä ruutua.

   Tekstiobjektin tekstin väriksi vaihtuu tumma purppura. Valitsemasi väri näkyy Properties-ikkunassa heksadesimaalisena (16-kantaisena) lukuna. Useimpien ohjelmoijien ei tarvitse välittää näistä heksadesimaalikoodeista, mutta on kiinnostava nähdä, miten Visual Basic tallentaa tällaiset ominaisuudet ohjelmaan. Nyt olet valmis määrittämään viimeisen objektin ominaisuudet.

Kuvakehyksen ominaisuudet

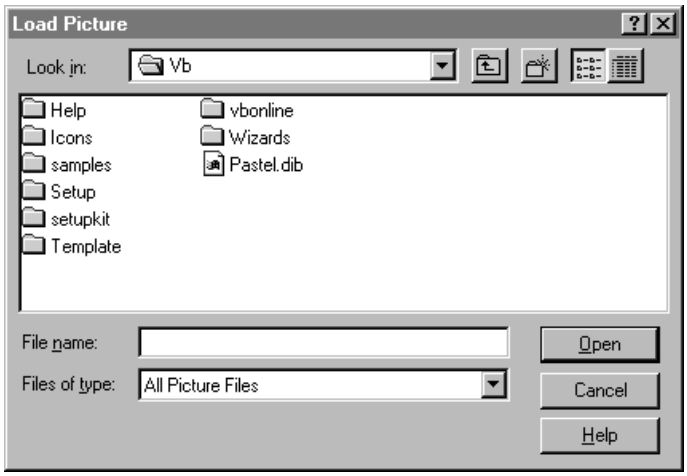
Kuvaobjektiin sijoitetaan kuva, joka esittää kolikkopinoa. Kuva tulee näkyviin, kun pelaaja voittaa eli kun ainakin yksi ohjelman näyttämistä satunnaisluvuista on seitsemän. Sinun pitää määrittää Stretch-ominaisuus, joka määrää kuvan koon tarkasti, Picture-ominaisuus, joka sisältää kuvaobjektiin luettavan kuvatiedoston nimen, ja Visible-ominaisuus, joka valitsee kuvaobjektin tilan ohjelman käynnistyessä.

Kuvakehyksen ominaisuuksien määrittäminen

1. Napsauta lomakkeen kuvaobjektia.
2. Napsauta Properties-ikkunan Stretch- eli venytysominaisuutta, napsauta avattavan luetteloruudun nuolipainiketta ja valitse True.

   Kun valitset Stretch-ominaisuudeksi True ennen kuvatiedoston avaamista, Visual Basic muuttaa kuvan koon tarkasti kuvaobjektin mittojen mukaiseksi. Tämä ominaisuus määritetään yleensä ennen Picture-ominaisuutta.
3. Kaksoisnapsauta Properties-ikkunan Picture- eli kuvaominaisuutta.

   Load Picture -valintaikkuna avautuu seuraavan kuvan näköisenä:



4. Siirry kansioon \Vb5Sbs napsauttamalla Load Picture -valintaikkunan Ylempi taso -painiketta.

   Näet kansion \Vb5Sbs alikansiot.
5. Kaksoisnapsauta kansiota Less01.

   Windows-metatiedosto Coins.wmf näkyy Load Picture -valintaikkunassa. Windows-metatiedosto sisältää graafisia objekteja, joiden kokoja voidaan muuttaa niin, että ne näyttävät hyviltä isoina tai pieninä.

6 Valitse tiedosto Coins.wmf ja napsauta Open.

Coins-tiedoston sisältö ilmestyy lomakkeen kuvakehykseen.

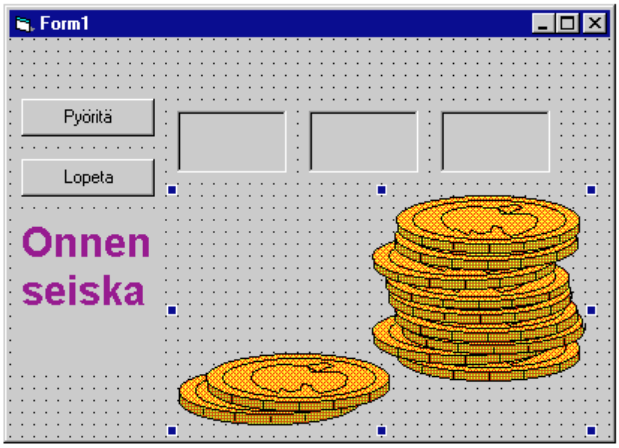
Muuta Visible-ominaisuudeksi False, jotta kolikot eivät ole näkyvissä ohjelman käynnistyessä. Ne muutetaan myöhemmin näkyviksi ohjelmalauseilla.

7 Napsauta Visible-ominaisuutta ja Visible-luetteloruudun nuolipainiketta.

Visible-ominaisuuden sallitut asetukset näkyvät luetteloruudussa.

8 Valitse False, jotta kuva ei näy ohjelman käynnistyessä.

Visible-ominaisuuden arvo on nyt False. Tämä vaikuttaa kuvakehyksen ulkoasuun ohjelman toimiessa, mutta ei ohjelmaa suunnitellessasi. Valmis lomake näyttää tällaiselta:



9 Telakoi Property-ikkuna kaksoisnapsauttamalla sen otsikkoriviä.

Ohjelmalauseiden kirjoittaminen

Nyt voit kirjoittaa Lucky Seven -ohjelman lauseet. Koska useimmat luomasi objektit ”tietävät”, miten niiden tulee toimia ohjelman käynnistyttyä, ne ovat valmiit vastaanottamaan syöttötietoja käyttäjältä ja käsittelemään ne automaattisesti. Objektien sisäiset toiminnot ovat yksi Visual Basicin tehokkaimmista ominaisuuksista — kun objektit on sijoitettu lomakkeeseen ja niiden ominaisuudet määritetty, ne pystyvät toimimaan ilman muiden ohjelmalauseiden kirjoittamista. Lucky Seven -ohjelman ”ydin” eli käskyt, jotka laskevat

Ominaisuuksien lukeminen taulukoista

Tässä harjoituksessa olen kuvannut Lucky Seven -ohjelman ominaisuudet yksitellen. Seuraavissa harjoituksissa esitän ominaisuuksien määrittämisen vaiheet taulukoina, ellei niihin liity mitään erikoista. Alla on lueteltu Lucky Seven -ohjelman tähän mennessä määritetyt ominaisuudet taulukkona, jonka kaltaisia näet myöhemmin tässä kirjassa.

Objekti	Ominaisuus	Asetus
Command1	Caption	”Pyöritä”
Command2	Caption	”Lopeta”
Label1, Label2, Label3	BorderStyle	1 - Fixed Single
	Alignment	2 - Center
	Font	Times New Roman, Bold, 24 pistettä
	Caption	(Tyhjä)
Label4	Caption	“Onnen seiska”
	Font	Arial, Bold, 20 pistettä
	ForeColor	Tumma purppura (&H008000808)
Image1	Picture	”\Vb5Sbs\Less01\coins.wmf”
	Stretch	True
	Visible	False

satunnaisluvut, näyttävät ne tekstikehyksissä ja tarkistavat, onko jokin niistä seitsemän — ovat vielä kirjoittamatta. Ohjelman toimintalogiikka on mahdollista muodostaa vain kirjoittamalla ohjelmalauseita, jotka määräävät tarkoin, mitä ohjelman pitää tehdä missäkin vaiheessa. Koska ohjelmaa ohjataan Pyöritä- ja Lopeta-painikkeilla, liitä pelin ohjelmalauseet niihin. *Code-ikkuna* on ohjelmointiympäristön erikoisikkuna, jossa kirjoitetaan ja muokataan Visual Basic -ohjelmalauseita.

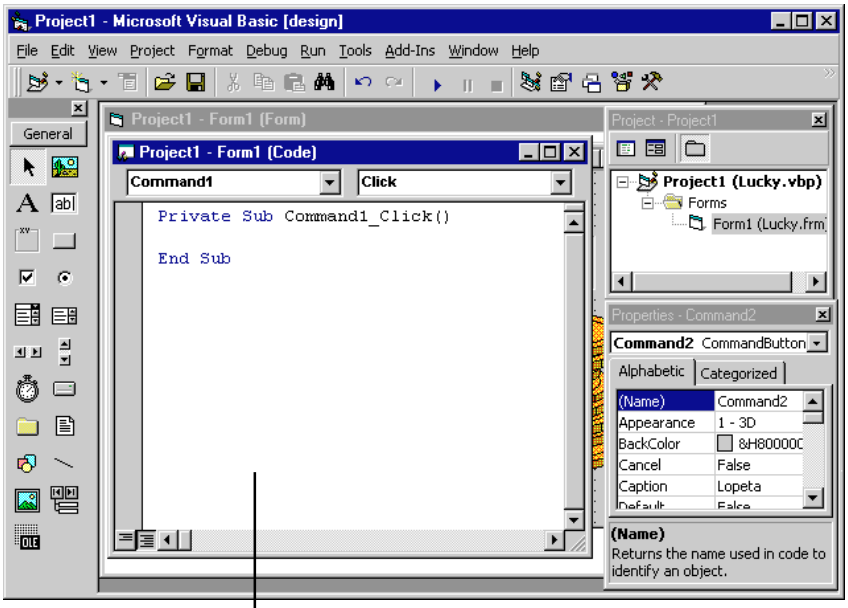
Ohjelmalauseet kirjoitetaan Code-ikkunaan.

Seuraavissa vaiheissa kirjoitetaan Lucky Seven -ohjelman lauseet Code-ikkunaan.

Code-ikkunan käyttö

1. Kaksoisnapsauta lomakkeen Lopeta-komentopainiketta.

   Kun kaksoisnapsautat objektia, näyttöön tulee seuraavan kuvan kaltainen Code-ikkuna.



Code-ikkuna

Jos ikkuna on pienempi kuin kuvassa, muuta sen kokoa hiirellä. Tarkalla koolla ei ole väliä, koska Code-ikkunassa on vierityspalkit, joilla voidaan tuoda esiin pitkienkin ohjelmalauseiden kaikki osat.

Code-ikkunassa on ohjelmalauseet, jotka osoittavat tämän Visual Basic -aliohjelman alun ja lopun. Aliohjelmaa kutsutaan myös *tapahtuma-aliohjelmaksi*, koska sen lauseet liittyvät käyttöliittymän tiettyä osaa koskeviin tapahtumiin. Se näyttää tällaiselta:

```
Private Sub Command2_Click()

End Sub
```

Aliohjelman runko on aina tällaisten rivien välissä ja suoritetaan, kun käyttäjä valitsee käyttöliittymän osan, johon aliohjelma on kytketty. Tässä tapauksessa tapahtuma on hiiren napsautus, mutta kuten myöhemmin näet, se voisi olla jokin muukin toiminto.

2 Kirjoita **End** ja paina ALANUOLINÄPPÄINTÄ.

End-lause lopettaa ohjelman suorituksen.

Lausetta kirjoittaessasi kirjaimet tulevat Code-ikkunaan mustina. Painettuasi ALANUOLINÄPPÄINTÄ (voisit myös painaa ENTER-näppäintä tai napsauttaa ohjelman jotakin toista riviä), lause muuttuu siniseksi. Tämä on merkki siitä, että Visual Basic on tunnistanut sen ohjelman sallituksi lauseeksi eli *avainsanaksi*. End on ohjelmalause, jolla ohjelma lopetetaan ja sen ikkuna poistetaan näytöstä. Visual Basic -ohjelmointijärjestelmä sisältää satoja tämänkaltaisia yksikäsitteisiä avainsanoja sekä niihin liittyviä

operaattoreita ja symboleita. Nämä on kirjoitettava ja eroteltava toisistaan välilyönnein täsmälleen oikein, jotta Visual Basic -kääntäjä tunnistaisi ne.

HUOMAA Sääntöjä, jotka koskevat ohjelman avainsanojen oikeinkirjoitusta, järjestystä ja välimerkkejä, sanotaan *kieliopiksi* tai *syntaksiksi*.

3 Siirrä kohdistin End-sanan sisältävän rivin alkuun ja paina VÄLINÄPPÄINTÄ neljästi.

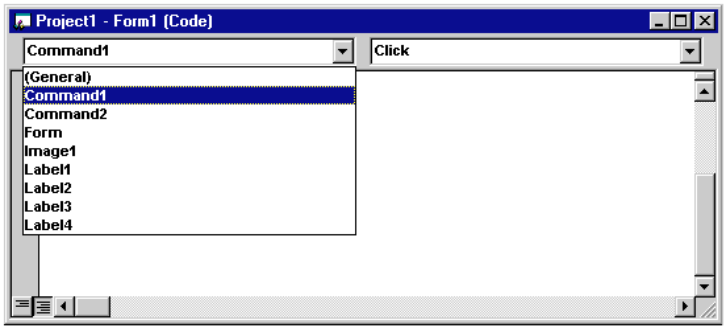
Tämä sisennys siirtää End-lauseen neljä merkkiä oikealle, jotta se erottuu Private Sub- ja End Sub- lauseista. Sisentäminen on yksi tässä kirjassa noudatettavista ohjelmointisäännöistä, joiden tarkoitus on parantaa ohjelmien luettavuutta. Sääntöjä, jotka koskevat ohjelmakäskyjen järjestämistä ohjelmaan, kutsutaan usein *ohjelmointityyliksi*.

Kun olet kirjoittanut Lopeta-painikkeeseen liittyvät ohjelmalauseet, määritä Pyöritä-painikkeen lauseet. Nämä ovat monimutkaisempia ja havainnollistavat ohjelman kielioppia ja tyyliä hieman paremmin. Kerron seuraavista ohjelmalauseista myöhemmin lisää, joten sinun ei tarvitse vielä ymmärtää niitä kaikkia. Keskity vain ohjelmalauseiden yleiseen rakenteeseen ja kirjoita lauseet täsmälleen sellaisina kuin esitän ne. (Visual Basic on hyvin tarkka siitä, että avainsanat ja operaattorit on kirjoitettu virheettömästi ja oikeaan järjestykseen.)

Spin-painikkeen lauseiden kirjoittaminen

1 Avaa Code-ikkunan objektiluettelo.

Lucky Seven -ohjelman objektit näkyvät luettelossa tällaisina:



2 Napsauta luettelon Command1-objektia.

Ensimmäisen aliohjelman yläpuolelle ilmestyy aliohjelma, joka liittyy Command1-painikkeeseen. Visual Basic näyttää tavallisesti lomakkeen kaikki tapahtuma-aliohjelmat yhdessä ikkunassa, jotta voit helposti siirtyä aliohjelmasta toiseen (ne on eroteltu vaakaviivoin). Voit katsella kutakin aliohjelmaa omassa ikkunassaan napsauttamalla Code-ikkunan vasemmassa alakulmassa olevaa pientä Aliohjelmanäkymä-painiketta. Kun haluat jälleen nähdä kaikki aliohjelmat samassa ikkunassa, napsauta Moduulinäkymä-painiketta (Aliohjelmanäkymä-painikkeen oikealla puolella).

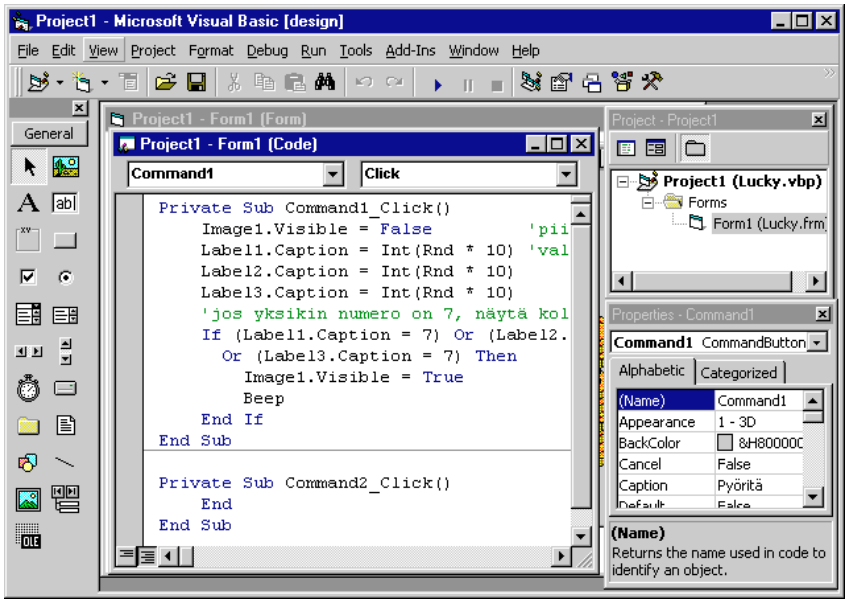
Vaikka vaihdoit painikkeen tekstiksi ”Pyöritä”, sen nimi ohjelmassa on edelleen Command1. (Käyttöliittymän osan nimi ja teksti voivat olla erilaiset, jos ohjelmoija näin haluaa.) Jokaiseen objektiin voi liittyä useita aliohjelmia; yksi kutakin objektin tunnistamaa tapahtumaa varten. Olemme nyt kiinnostuneita napsautustapahtumasta, koska käyttäjä napsauttaa Pyöritä- ja Lopeta-painikkeita pelatessaan peliä.

3 Kirjoita seuraavat ohjelmarivit Private Sub- ja End Sub -lauseiden väliin. Paina kunkin rivin jälkeen ENTER ja kirjoita lauseet täsmälleen sellaisina kuin ne näkyvät tässä. (Code-ikkuna vierittyy vasemmalle, kun kirjoitat pitkiä rivejä.) Jos teet virheen (Visual Basic merkitsee sen tavallisesti punaisella), poista virheellinen lause ja yritä uudelleen.

```
Image1.Visible = False          ‘ piilota     kolikot
Label1.Caption = Int(Rnd * 10) ‘ valitse      numerot
Label2.Caption = Int(Rnd * 10)
Label3.Caption = Int(Rnd * 10)
‘ jos yksikin numero on 7, näytä kolikot ja piippaa
If (Label1.Caption = 7) Or (Label2.Caption = 7) _
  Or (Label3.Caption = 7) Then
    Image1.Visible = True
    Beep
End If
```

HUOMAA Kun kirjoitat ohjelmaa, Visual Basic muotoilee tekstin ja näyttää ohjelman eri osat värillisinä, jotta ne tulevat selvemmin esiin. Kun alat kirjoittaa ominaisuutta, Visual Basic näyttää kyseisen objektin ominaisuudet luetteloruudussa. Kaksoisnapsauta ominaisuutta tai kirjoita se itse. Jos Visual Basic näyttää virheilmoituksen, olet ehkä kirjoittanut jonkin ohjelmalauseen väärin. Vertaa riviä tämän kirjan tekstiin, tee tarpeelliset korjaukset ja jatka kirjoittamista. (Voit myös poistaa rivin ja kirjoittaa sen kokonaan uudelleen.) Visual Basicin valikkokomennot ja muokkausnäppäimet ovat samankaltaisia kuin muissakin Windows-sovelluksissa. Löydät lisätietoja Visual Basicin ohjeista.

Kun olet valmis, Code-ikkunan pitäisi näyttää tällaiselta:



Aliohjelman Command1_Click tarkasteleminen

Aliohjelma Command1_Click on Lucky Seven -ohjelman ydin.

Aliohjelma Command1_Click suoritetaan, kun käyttäjä napsauttaa valintaikkunan Pyöritä-painiketta. Se sisältää eräitä aika monimutkaisia lauseita, jotka saattavat näyttää sekavilta, koska en ole vielä esitellyt niitä muodollisesti. Jos tarkastelet aliohjelmaa lähemmin, huomaat varmaankin muutamia tuttuja seikkoja. Tutki hieman aliohjelman sisältöä saadaksesi käsityksen siitä, millaisia ohjelmia joudut tekemään myöhemmin tässä kirjassa. Jos et välitä eritellä aliohjelmaa, voit aivan hyvin siirtyä seuraavaan kohtaan ”Ohjelman tallentaminen”.

Aliohjelma Command1_Click suorittaa kolme tehtävää: se piilottaa kolikkopinon, muodostaa kolme satunnaislukua tekstikehyksiin sijoitettavaksi ja näyttää kolikkopinon, jos ainakin yksi satunnaisluvuista on seitsemän. Tarkastellaan näitä tehtäviä yksitellen.

Ensimmäisen tehtävän hoitaa rivi

```
Image1.Visible = False          ‘ piilota  kolikot
```

Tällä rivillä on kaksi osaa: ohjelmalause ja kommentti. Ohjelmalause (Image1.Visible = False) määrittää ensimmäisen kuvaobjektin (Image1) Visible-ominaisuuden arvoksi False (toisen kahdesta mahdollisesta arvosta). Muistanet, että asetit ominaisuuden arvoksi False aiemminkin käyttäen Properties-ikkunaa. Se on asetettava uudelleen ohjelmassa, koska ennen uutta peliä ikkunasta pitää poistaa kolikkopino, joka on voinut jäädä siihen edellisestä pelistä. Koska ominaisuutta

muutetaan suorituksen eikä suunnittelun aikana, se pitää määrittää ohjelmalauseella. Tämä on Visual Basicin kätevä ominaisuus, ja kerron siitä lisää luvussa 2.

Kommentit kuvaavat ohjelmalauseiden toimintaa.

Ensimmäisen rivin toinen osa (joka on kirjoitettu näytössä vihreällä) on nimeltään *kommentti*. Kommentit ovat ohjelmaan lisättyjä selittäviä huomautuksia, joiden alussa on puolilainausmerkki (') . Niillä ohjelmoija kuvaa, miten ohjelman tärkeät lauseet toimivat. Visual Basic ei välitä kommenteista suorittaessaan ohjelmaa; ne vain helpottavat ohjelman ymmärtämistä. Sinun kannattaa kirjoittaa ohjelmiisi runsaasti kommentteja, joissa kerrotaan "selväkielisesti", mistä on kysymys.

Kolme seuraavaa riviä tuottavat satunnaisluvut. Rnd-funktio palauttaa desimaalimuotoisen satunnaisluvun 0–1 ja Int-funktio kertoo tämän 10:llä ja pyöristää tuloksen alaspäin lähimpään kokonaislukuun. Lopputulos on siten satunnaisluku 0–9. Saadut luvut sijoitetaan lomakkeen kolmen ensimmäisen tekstiobjektin Caption-ominaisuuksiin. Sijoituslause aiheuttaa sen, että numerot näkyvät valintaikkunassa lihavoituina, 24 pisteen kokoisina ja Times New Roman -fontilla kirjoitettuina.

Ohjelman viimeinen lauseryhmä tarkistaa, onko ainakin yksi satunnaisluvuista seitsemän. Jos näin on, ohjelma tuo kolikkopinon näkyviin ja antaa äänimerkin osoitukseksi pelin voittamisesta. Joka kerta kun käyttäjä napsauttaa Spin-painiketta, Command1_Click-aliohjelma käynnistyy ja sen sisältämät lauseet suoritetaan.

Ohjelman tallentaminen

Voit tallentaa ohjelman milloin vain ohjelmointityön aikana.

Nyt kun olet saanut Lucky Seven -ohjelman valmiiksi, sinun pitäisi tallentaa se levylle. Visual Basic sijoittaa lomakkeen käskyt ja objektit yhteen tiedostoon ja projektin osien "pakkauslistan" toiseen tiedostoon (osat on lueteltu Project-ikkunassa). Voit käyttää näitä osatiedostoja muissa ohjelmointiprojekteissa valitsemalla Project-valikosta komennon Add File. Ohjelma tallennetaan Visual Basicissa valitsemalla File-valikosta komento Save Project As tai napsauttamalla työkalurivin Tallenna projekti -painiketta.

Lucky Seven -ohjelman tallentaminen

1. Valitse File-valikosta komento Save Project As.

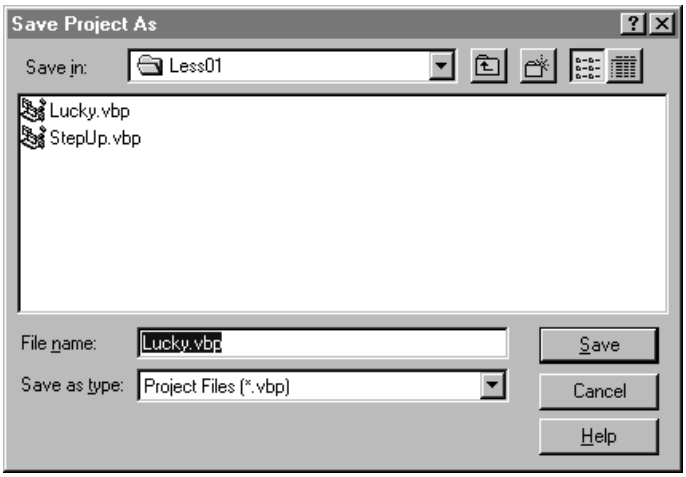
   Save File As -valintaikkuna avautuu ja kysyy lomakkeen nimen ja tallennuspaikan.

2. Valitse valintaikkunasta kansio Less01, ellei sitä ole vielä valittu.

   Tallenna projektitiedostosi harjoituskansioon, jonka tähän kirjaan liittyvä asennusohjelma loi kiintolevyllesi. Voit halutessasi valita jonkin muun hakemiston.

3 Kirjoita File Name -tekstiruutuun **MyLucky** ja paina ENTER.

Lucky Seven -lomake tallentuu nimellä MyLucky.frm. Sitten Save Project As -valintaikkuna avautuu:



HUOMAA Suosittelen, että tallennat tämän kirjan harjoituksissa luomasi projektit käyttäen etuliitettä My. Näin voit tarkistaa, mitkä harjoitukset olet tehnyt, sekä säilyttää alkuperäiset harjoitustiedostot. Tutki alkuperäisiä tiedostoja, jos ongelmia ilmenee.

4 Kirjoita **MyLucky** ja paina ENTER.

Lucky Seven -projekti tallentuu nimellä MyLucky.vbp. Jos haluat lukea tämän projektin myöhemmin muistiin, valitse File-valikosta komento Open Project ja valitse Open Project -valintaikkunasta MyLucky. Voit valita äskettäin käsitellyn projektin myös File-valikon alareunasta.

Valmis Lucky-ohjelma on kansiossa \Vb5Sbs\Less01.

Olet valmis käynnistämään ensimmäisen todellisen ohjelmasi. Kun haluat suorittaa Visual Basic -ohjelman ohjelmointiympäristössä, valitse Run-valikosta komento Start, napsauta työkalurivin Käynnistä-painiketta tai paina F5. Kokeile Lucky Seven -ohjelman käynnistämistä nyt. Jos Visual Basic näyttää virheilmoituksen, ohjelmassa on ehkä edelleen kirjoitusvirhe. Yritä korjata se vertaamalla edellä esitettyä ohjelmaa kirjoittamaasi. Voit myös lukea levyltä Lucky-ohjelman ja käynnistää sen.

Käynnistä-painike

Ohjelman suorittaminen

1 Napsauta työkalurivin Käynnistä-painiketta.

Lucky Seven -ohjelma käynnistyy ohjelmointiympäristössä. Käyttöliittymä tulee näkyviin juuri sellaisena kuin oli tarkoitus.

2 Napsauta Pyöritä-painiketta.

Ohjelma valitsee kolme satunnaislukua ja näyttää ne ikkunan tekstikehyksissä näin:

Koska ensimmäinen numero on seitsemän, ikkunaan tulee kolikkopinon kuva, ja tietokone antaa äänimerkin. (Ääni riippuu Windows 95:n Ohjauspaneelin asetuksesta.) Sinä voitit!

3 Napsauta Pyöritä-painiketta parikymmentä kertaa ja seuraa pelin tuloksia.

Voitat keskimäärin puolet peleistä, joten voittomahdollisuutesi ovat oikein hyvät. Laskennallinen todennäköisyys on kolme kymmenestä, mutta sinulla on ehkä aloittelijan tuuria. Haluat varmaan tehdä pelistä vaikeamman niin, että voitat vain, jos luku seitsemän esiintyy kahdesti tai kolmesti. Opit tekemään tällaisia muutoksia, kun olen kertonut moduuleista ja julkisista muuttujista harjoituksessa 9.

4 Kun olet pelannut tarpeeksesi, napsauta Lopeta-painiketta.

Ohjelma pysähtyy, ja ohjelmointiympäristön osat tulevat jälleen näkyviin.

Suoritettavan tiedoston muodostaminen

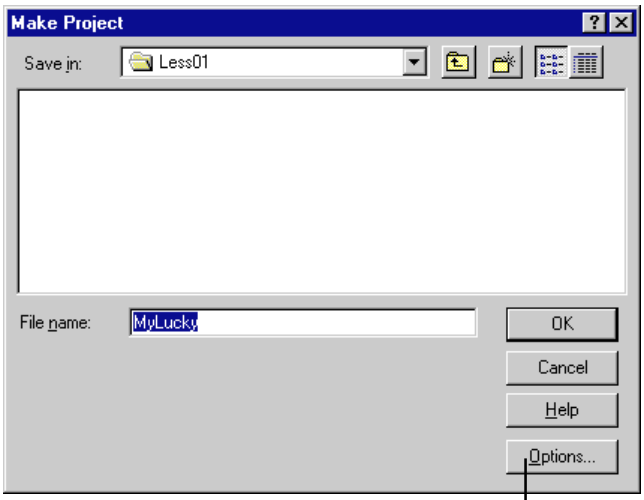
.exe-tiedosto toimii Windows 95:ssä ja Windows NT:ssä.

Viimeinen tehtäväsi tässä harjoituksessa on päättää ohjelman kehittely luomalla Windows-sovellus eli *suoritettava* tiedosto. Visual Basicilla tehtyjen sovellustiedostojen tarkenne on .exe, ja ne toimivat missä tahansa järjestelmässä, jossa on Windows 95 tai Windows NT 3.51 tai uudempi sekä tarvittavat tukitiedostot. Visual Basic asentaa tukitiedostot — myös dynaamiset linkkikirjastot ja mukautetut objektit — automaattisesti. Jos aiot jakaa sovelluksiasi, katso lisätietoja *Visual Basic Programmer's Guide* -oppaasta. Yritä nyt luoda tiedosto MyLucky.exe.

Suoritettavan tiedoston luominen

1 Valitse File-valikosta komento Make MyLucky.exe. (Visual Basic lisää komentoon automaattisesti ohjelman nimen.)

Make Project -valintaikkuna avautuu tällaisena:



Asetukset-painiketta napsauttamalla saat esiin lisää kääntäjän asetuksia

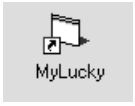
Valintaikkuna sisältää tekstiruutuja ja luetteloruutuja, joilla voit määrittää suoritettavan tiedoston nimen ja sijainnin levyllä. Siinä on myös Options-painike, jolla avautuu Project Properties -valintaikkuna ohjelmatiedoston kuvakkeen ja versiotietojen määrittämistä varten. Oletuksen mukaan Visual Basic ehdottaa tiedoston sijoituspaikaksi kansiota Less01.

VIHJE Project Properties -valintaikkunassa (avataan Project-valikosta) on Visual Basic version 5 uutena ominaisuutena Compile-välilehti, jossa määritetään ohjelman kääntämiseen liittyviä toimintoja. Voit esimerkiksi optimoida ohjelman niin, että siitä tulee nopea ja tehokas tai pieni ja tiivis, etsiä virheitä ja valita muita erikoistoimintoja. Kun Visual Basiciin on lisätty nämä edistykselliset ominaisuudet, ohjelmoijalla on käytettävissään työkalut, jotka ovat perinteisesti kuuluneet vain kaikkein tehokkaimpiin kääntäjiin, kuten Microsoft Visual C++:aan.

2. Hyväksy tiedoston oletusnimi ja -sijainti levyllä napsauttamalla OK.

   Visual Basic luo suoritettavan tiedoston määrittämääsi paikkaan.

Kun haluat käynnistää tämän ohjelman Windows 95:ssä, valitse Käynnistä-valikosta (Start) Suorita (Run) tai napsauta tiedoston nimeä Windows Resurssienhallinnassa (Explorer). Voit myös lisätä Windowsin työpöydälle ohjelman pikakuvakkeen napsauttamalla työpöytää hiiren oikealla painikkeella ja valitsemalla Uusi (New) ja Pikakuvake (Shortcut). Kun Windows kysyy sovellustiedoston sijainnin, napsauta Selaa (Browse) ja valitse suoritettava MyLucky-tiedosto kansiosta \Vb5Sbs\Less01. Napsauta Avaa (Open), Seuraava (Next) ja Valmis (Finish), jotta Windows lisää työpöydälle kuvakkeen, jota kaksoisnapsauttamalla ohjelma käynnistyy. Pikakuvake näyttää tällaiselta:



Nyt olet suorittanut kaikki tähän harjoitukseen kuuluvat tehtävät. Voit poistua ohjelmointiympäristöstä alla kuvatulla tavalla.

Visual Basicin lopetus

1. Tallenna ohjelmaan tekemäsi muutokset.
2. Valitse File-valikosta komento Exit.

   Visual Basic -ohjelman suoritus päättyy.

Lisätietoja: Lisäysten tekeminen ohjelmaan

Voit käynnistää Visual Basicin uudelleen milloin vain ja muokata levylle tallentamaasi ohjelmointiprojektia. Käynnistä Visual Basic nyt ja lisää Lucky Seven -ohjelmaan erityislause nimeltä Randomize.

Lucky Seven -ohjelman lukeminen muistiin

1. Napsauta Windowsin tehtäväpalkin Käynnistä-painiketta (Start), valitse Ohjelmat (Programs) ja Visual Basic 5.0 ja napsauta Visual Basic 5.0 -ohjelmakuvaketta.
2. Napsauta New Project -valintaikkunan Recent-välilehteä.

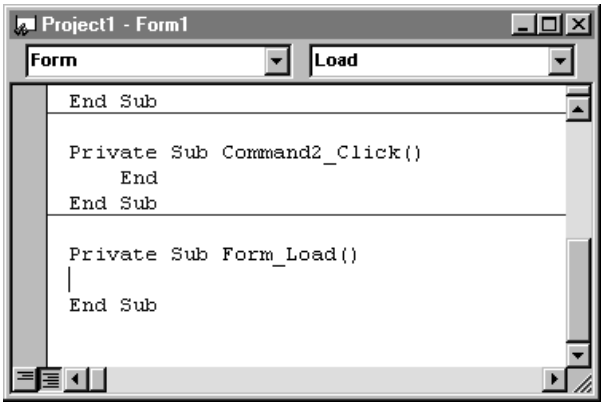
   Näet luetteloruudussa viimeksi käsittelemiesi projektien nimet. Koska olet juuri lopettanut Lucky Seven -ohjelman muokkaamisen, luettelon ensimmäisen projektin pitäisi olla MyLucky.
3. Lataa ohjelma levyltä kaksoisnapsauttamalla nimeä MyLucky.

   Lucky Seven -ohjelma latautuu levyltä, ja MyLucky-lomake ilmestyy ikkunaan. Ellet näe sitä, valitse MyLucky-lomake Project-ikkunasta ja napsauta Näytä objekti -painiketta.

Lisää Randomize-lause aliohjelmaan Form_load. Tämä on erikoisaliohjelma, joka liittyy lomakkeeseen ja suoritetaan aina ohjelman käynnistyessä.

4 Kaksoisnapsauta lomaketta (ei objektia), jotta näet aliohjelman Form_load.

Aliohjelma ilmestyy Code-ikkunaan seuraavan kuvan näköisenä:



5 Paina VÄLINÄPPÄINTÄ neljästi, kirjoita **Randomize** ja paina ALANUOLINÄPPÄINTÄ.

Ohjelmassa on nyt Randomize-lause, joka suoritetaan aina ohjelman käynnistyessä. Se lukee ajan järjestelmäkellosta ja muodostaa todella satunnaisen alkuarvon eli ”siemenluvun” Rnd-lauseelle, joka on aliohjelmassa Command1_Click. Et ehkä huomannut, että ilman Randomize-lausetta ohjelma tuottaa aina samat satunnaisluvut. Kun Randomize-lause on lisätty, tulokset ovat satunnaisia tai eivät ainakaan noudata mitään tunnistettavaa kaavaa.

6 Kokeile ohjelman uutta versiota ja tallenna projekti levylle. Jos aiot käyttää ohjelmaa usein, sinun kannattaa ehkä tehdä uusi .exe-tiedosto. Visual Basic ei päivitä suoritettavia tiedostoja automaattisesti, kun lähdetiedostoa on muutettu.

Jos haluat jatkaa seuraavaan harjoitukseen

➤ Jätä Visual Basic käyntiin ja siirry harjoitukseen 2.

Jos haluat lopettaa Visual Basicin

➤ Valitse File-valikosta komento Exit.

Jos näyttöön tulee Save-valintaikkuna, napsauta Yes.

Harjoituksen tiivistelmä

Tehtävä	Ohje	Painike
Käynnistää Visual Basic	Napsauta tehtäväpalkin Käynnistä-painiketta (Start). Valitse Ohjelmat (Programs), Visual Basic 5.0 ja napsauta Visual Basic 5.0 -ohjelmakuvaketta.	
Näyttää painikkeen toiminto	Siirrä hiiren osoitin hetkeksi painikkeen päälle.	
Avata projekti	Valitse File-valikosta komento Open Project.	
Aloittaa uusi projekti	Valitse File-valikosta komento New Project.	
Suorittaa ohjelma	Napsauta työkalurivin Käynnistä-painiketta. *tai* Paina F5.	
Telakoida ohjelmointi-työkalu	Napsauta työkalun otsikkoriviä ja vedä työkalu toisen työkalun viereen, jotta se napsahtaa paikalleen. Jos haluat nähdä enemmän telakoidusta työkalusta, kaksoisnapsauta otsikko-riviä tai muuta työkalun kokoa hiirellä.	
Siirtää työkaluruutua	Vedä palettia hiirellä.	
Määrittää ominaisuuksia	Napsauta työkalurivin Properties-ikkuna-painiketta, jotta Properties-ikkuna avautuu (ellei se ole avoinna) ja kaksoisnapsauta sen otsikkoriviä. Avaa objektiluettelo, jotta näet lomakkeen käyttöliittymän osat, napsauta objektia, jonka ominaisuudet haluat määrittää, ja valitse sopivat ominaisuusasetukset Properties-luettelosta.	
Avata Project-ikkuna	Napsauta työkalurivin Project--ikkuna-painiketta ja kaksoisnapsauta Project-ikkunan otsikkoriviä.	
Luoda käyttöliittymä	Lisää lomakkeeseen objekteja työkalu-ruudun työkaluilla ja määritä niiden ominaisuudet. Muuta lomakkeen ja objektien kokoa tarvittaessa.	

Tehtävä	Ohje	Painike
Siirtää objektia	Vedä objektia lomakkeessa hiirellä.	
Muuttaa objektin kokoa	Valitse objekti ja vedä objektin siihen osaan liitettyä valintakahvaa, jonka kokoa haluat muuttaa.	
Poistaa objekti	Valitse objekti ja paina DEL.	
Avata Code-ikkuna	Kaksoisnapsauta lomakkeessa olevaa objektia (tai itse lomaketta).	
	tai	
	Napsauta Project-ikkunan Näytä lauseet -painiketta, kun lomakkeen tai moduulin nimi on valittu Project-ikkunasta.	
Kirjoittaa ohjelmalauseita	Kirjoita Code-ikkunaan Visual Basic -ohjelmalauseet, jotka haluat liittää valitsemaasi objektiin.	
Tallentaa ohjelma	Valitse File-valikosta komento Save Project As.	
	tai	
	Napsauta työkalurivin Tallenna projekti -painiketta.	
Luoda .exe-tiedosto	Valitse File-valikosta komento Make *tiedosto*.exe.	
Lopettaa Visual Basic	Valitse File-valikosta komento Exit.	
Ladata projekti muistiin	Valitse File-valikosta komento Open Project.	
	tai	
	Kaksoisnapsauta tiedoston nimeä New Project -valintaikkunan Recent-välilehdessä.	

Kun tarvitset tietoja aiheesta	Valitse Visual Basicin Help-valikosta Microsoft Visual Basic Help Topics, valitse Index-välilehti ja...
Objektien käyttö	Kirjoita **controls, overview** ja paina ENTER.
Ominaisuuksien määrittäminen	Kirjoita **properties window, elements** ja paina ENTER.
Ohjelmalauseiden kirjoittaminen	Kirjoita **code, writing and editing** ja paina ENTER.

Tietoja seuraavasta harjoituksesta

Seuraavassa harjoituksessa tutustut tarkemmin työkaluruudussa oleviin käyttöliittymän osiin. Opit lisäämään käyttöliittymän osia lomakkeisiin, viittaamaan niihin ohjelmalauseissa ja käyttämään niitä tehokkaasti ohjelmissa.

Harjoitus

2

Objektien käyttö

Arvioitu aika
55 min.

Tässä harjoituksessa opit

- käyttämään tekstiruutu- ja komentopainikeobjekteja luodessasi ”Hello World” -ohjelman
- etsimään kuvatiedostoja levyltä tiedostojärjestelmäobjekteilla ja kuva-objektilla
- käsittelemään syöttötietoja valintanappi-, valintaruutu- ja luetteloruutu-objekteilla
- käynnistämään Windows-ohjelman OLE-objektilla
- näyttämään Microsoft Access -tietokannan tietueet dataobjektilla
- asentamaan ActiveX-objekteja.

Kuten harjoituksessa 1 huomasit, Microsoft Visual Basicin työkalut ovat graafisia välineitä, joilla muodostetaan Visual Basic -ohjelman käyttöliittymä. Työkalut ovat ohjelmointiympäristön työkaluruudussa, ja niillä lisätään lomakkeeseen objekteja yksinkertaisesti napsauttamalla ja vetämällä hiirtä. Tässä harjoituksessa opit näyttämään tietoja tekstiruudussa, selaamaan järjestelmän levyasemia ja kansioita, käsittelemään syöttötietoja, käynnistämään Windows-sovelluksia ja tutkimaan tietokannan tietueita. Opetan sinut suunnittelemaan omia Visual Basic -sovelluksiasi ja kerron lisää objekteista, ominaisuuksista ja ohjelmalauseista. Esitän myös, miten voit lisätä työkaluruutuun ActiveX-objekteja (aiemmalta nimeltään OLE-objekteja) laajentaaksesi Visual Basicin ominaisuuksia.

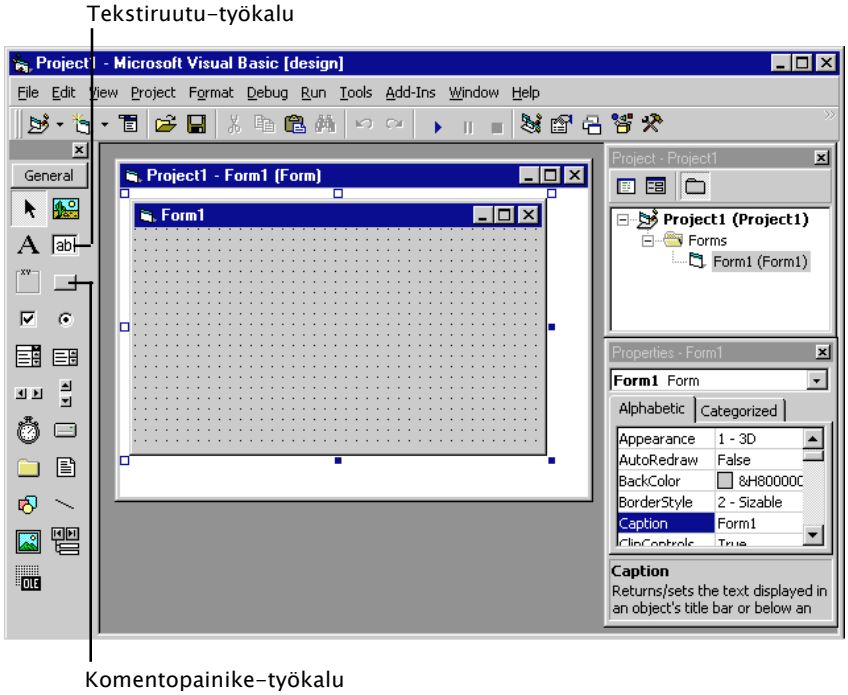
Objektien alkeita: "Hello World" -ohjelma

Ohjelmoinnin alkeisoppikirjojen perinteisiin kuuluu "Hello World" -ohjelma. Se on tapa havainnollistaa, miten yksinkertaisin mahdollinen ohjelma luodaan tietyllä ohjelmointikielellä ja suoritetaan. Merkkipohjaisen ohjelmoinnin aikoina Hello World oli tavallisesti kaksi- tai kolmirivinen ohjelma, joka kirjoitettiin editorilla ja käännettiin erillisellä kääntäjällä. Graafisten ohjelmointivälineiden tultua käyttöön tavanomainen Hello World on kasvanut monimutkaiseksi ohjelmaksi, jossa on kymmeniä rivejä ja jonka muodostamiseen tarvitaan monia työkaluja. Onneksi Hello World -ohjelman tekeminen Visual Basicilla on vieläkin aika helppoa. Saat aikaan täydellisen käyttöliittymän luomalla kaksi objektia, määrittämällä kaksi ominaisuutta ja kirjoittamalla yhden ohjelmalauseen. Kokeillaanpa.

Hello World -ohjelman luominen

1 Käynnistä Visual Basic ja luo tavanomainen 32-bittinen sovellus.

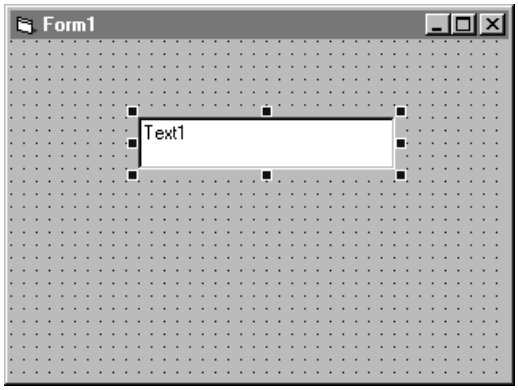
Visual Basic -ohjelmointiympäristö tulee näyttöön seuraavan kuvan mukaisena. Kuvaan on merkitty kaksi työkalua, joita tarvitset tässä harjoituksessa: tekstiruutu ja komentopainike.



Tekstiruutu-työkalu

2 Napsauta työkaluruudun Tekstiruutu-työkalua.

3 Siirrä hiiren osoitin lomakkeen keskelle (se muuttuu pieneksi ristiksi) ja piirrä tällainen tekstiruutu:



Tekstiruutuobjektia käytetään tekstin näyttämiseen tai syöttötietojen lukemiseen Visual Basic -ohjelmassa. Tekstiruutuobjektin toimintatapa riippuu siitä, millaiset ominaisuudet sille on annettu ja miten siihen viitataan ohjelmassa. Tämä pieni ohjelma näyttää tekstiruudussa tervehdyksen Hei kaikille!", kun komentopainiketta napsautetaan.

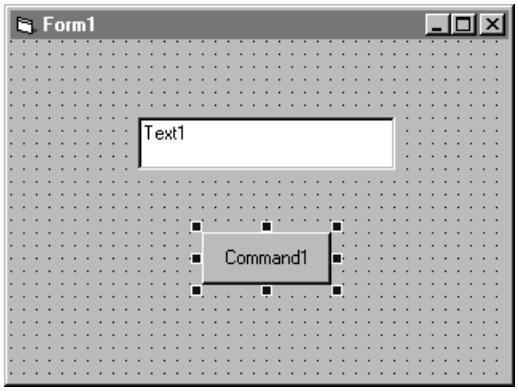
Lisää seuraavaksi komentopainike.

4 Napsauta työkaluruudun Komentopainike-työkalua.

5 Siirrä hiiren osoitin lomakkeessa tekstiruudun alapuolelle ja piirrä komentopainike.

Komentopainike-työkalu

Lomakkeen pitäisi näyttää suunnilleen tällaiselta:



Komentopainike on yksinkertaisin tapa lukea syöttötieto. Kun käyttäjä napsauttaa painiketta, hän haluaa ohjelman suorittavan tietyn tehtävän heti. Visual Basicin termein ilmaistuna käyttäjä aiheuttaa komentopainikkeella *tapahtuman*, joka ohjelman pitää käsitellä. Ohjelman tavanomaisia komentopainikkeita ovat OK, jota napsauttamalla käyttäjä hyväksyy joukon asetuksia ja osoittaa olevansa valmis jatkamaan, Peruuta eli Cancel, jolla käyttäjä hylkää joukon asetuksia, ja Lopeta eli Quit, jolla käyttäjä lopettaa ohjelman. Kussakin tapauksessa suunnittelijan pitää määrittää painikkeet niin, että ne toimivat odotetulla tavalla. Komentopainikkeen (ja kaikkien muidenkin objektien) ominaisuuksia muutetaan käyttämällä ominaisuusasetuksia tai viittaamalla objektiin ohjelmassa.

6 Määritä seuraavat tekstiruutu- ja komentopainikeobjektin ominaisuudet Properties-ikkunassa. Asetus (Tyhjä) tarkoittaa, että sinun tulee poistaa nykyinen asetus ja jättää ominaisuus tyhjäksi. Teksti, joka sinun pitää kirjoittaa, on ympäröity lainausmerkein. Älä kirjoita lainausmerkkejä.

Lisätietoja on harjoituksen 1 kohdassa "Ominaisuuksien määrittäminen".

Objekti	Ominaisuus	Asetus
Text1	Text	(Tyhjä)
Command1	Caption	"OK"

7 Kaksoisnapsauta OK-painiketta ja kirjoita seuraava lause Code-ikkunaan Private Sub- ja End Sub -lauseiden väliin:

Valmis Hello-ohjelma on kansiossa \Vb5Sbs\Less02.

```
Text1.Text = "Hei  kaikille!"
```

VIHJE Kun olet kirjoittanut objektin nimen Text1 ja pisteen, Visual Basic näyttää luetteloruudun, joka sisältää kaikki tekstiruutuobjektien ominaisuudet. Katso luetteloa, jos olet unohtanut ominaisuuden nimen. Voit valita luettelosta ominaisuuden kaksoisnapsauttamalla tai kirjoittaa sen itse. (Minä kirjoitan yleensä ominaisuuden nimen, ellen kokeile jotakin uutta toimintoa.)

Kirjoittamasi lause muuttaa tekstiruudun Text-ominaisuudeksi "Hei kaikille!", kun käyttäjä napsauttaa komentopainiketta ohjelman suorituksen aikana. Yhtäläisyysmerkki tarkoittaa sitä, että lainausmerkein ympäröity teksti sijoittuu Text1-objektin Text-ominaisuuteen. Tämä on esimerkki ominaisuuden muuttamisesta ohjelmallisesti, mikä on yksi yleisimpiä ohjelmalauseiden käyttötarkoituksia Visual Basicissa. Koska yllä esitetty lause on tapahtuma-aliohjelmassa, joka käynnistyy Command1-painiketta napsautettaessa, ominaisuusasetus (ja siten tekstiruudun sisältö) muuttuu heti käyttäjän napsautettua komentopainiketta.

Määritä Form Layout -ikkunassa, mihin ohjelman ikkuna sijoittuu suorituksen aikana.

8 Määritä Form Layout -ikkunan avulla, mihin kohtaan näyttöä ohjelman ikkuna avautuu suorituksen aikana. Ellei Form Layout -ikkunaa näy, valitse View-valikosta komento Form Layout Window.

Ohjelman ikkuna sijoittuu tavallisesti näytön vasempaan yläkulmaan, mutta voit muuttaa tätä asetusta vetämällä pientä lomakekuvaketta Form Layout -ikkunassa. Tämä mahdollisuus on hyödyllinen varsinkin sellaisessa ohjelmassa, joka avaa useita ikkunoita.

Nyt voit suorittaa Hello World -ohjelman ja tallentaa sen levylle.

Hello World -ohjelman suorittaminen

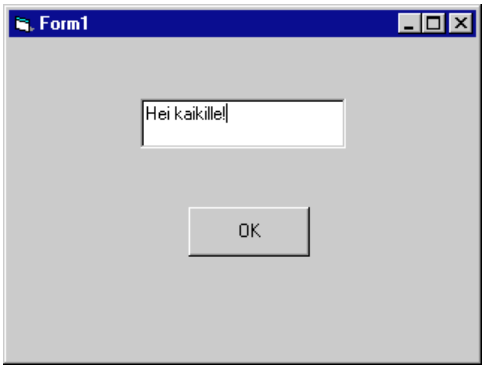
Käynnistä-painike

1 Napsauta työkalurivin Käynnistä-painiketta.

Hello World -ohjelma käynnistyy Visual Basic -ohjelmointiympäristön alaisuudessa.

2 Napsauta OK-painiketta.

Ohjelma näyttää tekstiruudussa tervehdyksen ”Hei kaikille!”, kuten tästä kuvasta näkyy:



Kun napsautit OK-painiketta, ohjelma muutti Text1-tekstiruudun Text-ominaisuudeksi ”Hei kaikille!”, ja tekstiruutuun ilmestyi vastaava teksti. Jos ohjelma ei toimi näin, toista edellisessä kohdassa ”Hello World -ohjelman luominen” esitetyt vaiheet ja käännä ohjelma uudelleen. Olet ehkä määrittänyt ominaisuuden väärin tai tehnyt ohjelmassa kirjoitusvirheen. (Visual Basic merkitsee kielioppivirheet punaisella.)

Lopeta-painike

3 Lopeta ohjelma napsauttamalla työkalurivin Lopeta-painiketta.

Voit myös napsauttaa ohjelman ikkunan sulkemispainiketta.

4 Valitse File-valikosta komento Save Project As.

5 Valitse kansio \Vb5Sbs\Less02, kirjoita **MyHello** ja napsauta Save.

Visual Basic tallentaa lomakkeen levylle nimellä MyHello.frm. Visual Basic tallentaa lomakkeet ja projektitiedostot erikseen, jotta voit käyttää lomakkeita ja aliohjelmia uudelleen myöhemmissä ohjelmointiprojekteissa tarvitsematta muodostaa niitä aina uudelleen.

Kun olet kirjoittanut lomakkeen nimen, Visual Basic kysyy nimen projektitiedostolle, jonka perusteella se muodostaa ohjelman. Visual Basic -projektitiedoston tarkenne on .vbp.

6 Kirjoita uudelleen **MyHello** ja napsauta Save.

Visual Basic tallentaa projektin levylle nimellä MyHello.vbp. Kun haluat avata ohjelman myöhemmin, valitse tämä tiedosto File-valikon komennolla Open Project. Visual Basic lukee kaikki projektiluettelossa mainitut tiedostot muistiin.

Onnittelut — olet nyt yksi monista ohjelmoijista, jotka ovat kirjoittaneet Hello World -ohjelman. Tutkitaan seuraavaksi muita objekteja.

Tiedostojärjestelmäobjektien käyttö

Visual Basicissa on kolme erittäin hyödyllistä objektia, joilla käsitellään tiedostojärjestelmää. Nämä ovat *levyasemaluetteloruutu*, jolla selataan järjestelmän levyasemia, *hakemistoluetteloruutu*, jolla selataan valitun aseman kansioita, ja *tiedostoluetteloruutu*, josta valitaan kansiossa oleva tiedosto. Seuraavassa harjoituksessa muodostat näistä objekteista Browser-nimisen ohjelman, jolla etsitään ja katsellaan grafiikkatiedostoja.

HUOMAA Ohjelmassa tarvitaan myös kuvaobjekti. Sillä voidaan näyttää kuudentyyppisiä grafiikkatiedostoja: bittikarttoja (.bmp), kooltaan muutettavia Windows-metatiedostoja (.wmf), kuvakkeita (.ico), osoittimia (.cur), JPEG-tiedostoja (.jpg) ja GIF-tiedostoja (.gif).

Browser-ohjelma

Browser-ohjelmassa on kolme tiedostojärjestelmäobjektia, kuvaobjekti ja useita ohjelmalauseita, joista muodostuu grafiikkatiedostojen selausohjelma. Kun olet saanut ohjelman valmiiksi, voit käyttää sitä päivittäisessä työssäsi, kun etsit grafiikkatiedostoja levykkeeltä, kiintolevyltä, verkkoasemasta tai CD-levyasemasta. Tiedostojärjestelmäobjektit tukevat kaikkia näitä asematyyppejä.

Browser-ohjelman luominen

1 Valitse File-valikosta komento New Project ja luo tavallinen suoritettava tiedosto napsauttamalla OK.

Hello World -ohjelma katoaa, ja näyttöön tulee tyhjä lomake. Voit tallentaa Hello World -ohjelmaan tekemäsi tallentamattomat muutokset ennen kuin se suljetaan.

2 Valitse Tools-valikosta komento Options ja napsauta Editor-välilehteä. Jos Require Variable Declaration -valintaruudussa on rasti, poista se napsauttamalla ruutua. (Tätä asetusta kuvataan harjoituksessa 4.) Napsauta OK.

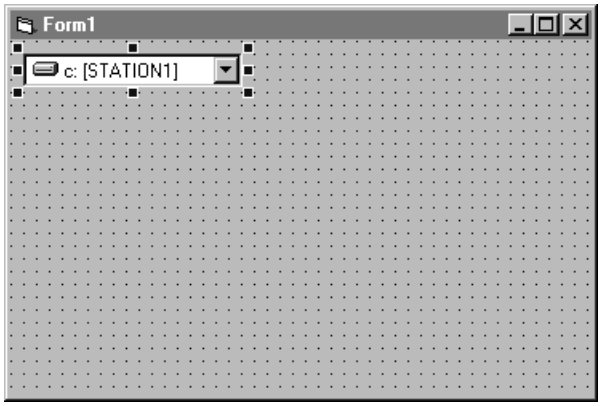
3 Suurenna lomaketta, jotta siinä on tilaa tiedostojärjestelmäobjekteille ja suurehkolle ikkunalle kuvien katselemista varten.

Lomakkeen ympärille ilmestyy vierityspalkit, joilla voit tuoda esiin haluamasi osan lomakkeesta, kun kehittelet ohjelmaa.

Levyasemaluettelo-työkalu

4 Napsauta työkaluruudun Levyasemaluettelo-työkalua.

5 Siirrä hiiren osoitin lomakkeen vasempaan yläkulmaan ja piirrä seuraavan kuvan mukainen levyasemaluetteloruutu:



Visual Basic sijoittaa luomaasi objektiin nykyisen levyaseman kirjaimen ja nimen. Näin käyttäjä voi nähdä valitun levyaseman nimen ohjelman suorituksen aikana, ja sinä näet samat tiedot, jotta voit muuttaa objektin koon sopivaksi. Jos kaikki tiedot eivät mahdu luetteloruutuun, suurenna sitä.

Hakemistoluettelo-työkalu

6 Napsauta työkaluruudun Hakemistoluettelo-työkalua ja lisää lomakkeeseen hakemistoluettelo levyasemaluettelon alapuolelle. Varaa siihen tilaa vähintään neljälle tai viidelle allekkaiselle kansiolle.

Hakemistoluetteloruutu on väline, jolla selataan tiedostojärjestelmän kansioita. Kun sijoitat tällaisen objektin lomakkeeseen, kansiot näkyvät heti samanlaisina kuin ohjelmaa suoritettaessa. Haluaisit varmaan napsautella kansioita jo nyt, mutta koska luetteloruutu ei ole aktiivinen, mitään ei tapahdu. Kansiot ovat näkyvissä vain siksi, että osaat muuttaa objektin koon sopivaksi.

Tiedostoluettelo-työkalu

7 Napsauta työkaluruudun Tiedostoluettelo-työkalua ja lisää lomakkeeseen tiedostoluettelo hakemistoluettelon alapuolelle. Varaa siihen tilaa vähintään neljälle tai viidelle allekkaiselle tiedostolle.

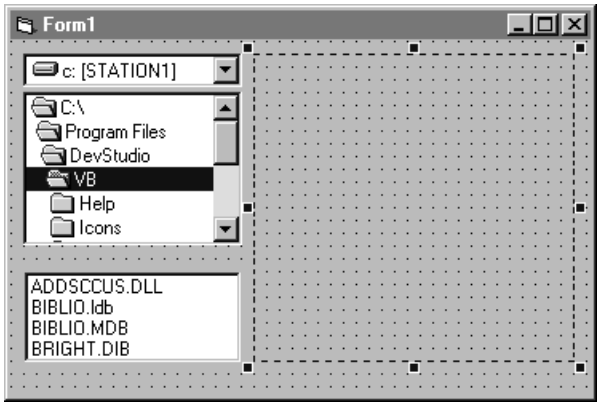
Tiedostoluettelosta käyttäjä voi valita minkä tahansa järjestelmään tallennetun tiedoston. Kun käyttäjä valitsee tiedoston, Visual Basic sijoittaa sen nimen tiedostoluettelo-objektin Filename-ominaisuuteen. Vastaavasti levyasemaluettelon Drive-ominaisuus ilmaisee käyttäjän valitseman levyaseman ja hakemistoluettelon Path-ominaisuus käyttäjän valitseman kansion. Käytät näitä kolmea ominaisuutta Browser-ohjelmassa, jotta se osaa avata käyttäjän valitseman grafiikkatiedoston. Tämä on tyypillinen tapa käyttää objekteja ja ominaisuuksia ohjelmassa: käyttäjä muuttaa objektin jotakin asetusta, muutos vaikuttaa ominaisuuteen ja ominaisuus käsitellään ohjelmalauseilla.

HUOMAA Drive-, Path- ja Filename-ominaisuudet ovat käytettävissä vain suorituksen aikana ja sisältävät arvoja, jotka muuttuvat tiedostojärjestelmäobjekteja käsiteltäessä. Et voi määrittää niiden arvoja Properties-ikkunassa.

Kuva-työkalu

8 Napsauta työkaluruudun Kuva-työkalua ja lisää lomakkeeseen suuri kuvakehys edellä piirtämiesi luetteloruutujen oikealle puolelle.

Kun olet lisännyt kuvaobjektin, lomakkeen pitäisi näyttää tällaiselta:



9 Määritä nyt seuraavat ominaisuudet Properties-ikkunassa:

Objekti	Ominaisuus	Asetus
File1	Pattern	*.bmp;*.wmf;*.ico
Image1	Stretch	True
Image1	BorderStyle	1 - Fixed Single

Tiedostoluettelo-objektin Pattern-ominaisuus on erittäin tärkeä tässä ohjelmassa. Se luettelee grafiikkatiedostomuodot, jotka Visual Basic pystyy esittämään kuvaobjektissa. Jos ominaisuus jätetään tyhjäksi,

tiedostoluettelossa näkyvät kansion kaikki tiedostot. Jos käyttäjä valitsee grafiikkatiedoston, jonka muotoa Visual Basic ei tue (esimerkiksi TIFF), aiheutuu ajonaikainen virhe eli ohjelma kaatuu. Tällaiset ongelmat on parasta estää hyvissä ajoin.

Lisää muutama ohjelmalause aliohjelmiin, jotka liittyvät tiedostojärjestelmäobjekteihin. Näitä aliohjelmia sanotaan *tapahtuma-aliohjelmiksi*, koska ne käynnistyvät, kun objektissa sattuu tietty tapahtuma, esimerkiksi hiiren napsautus.

Kaksoisnapsauta objektia, jotta näet sen oletustapahtuma-aliohjelman.

10 Kaksoisnapsauta lomakkeen levyasemaluettelo-objektia ja kirjoita seuraava lause Drive1_Change-aliohjelman Private Sub- ja End Sub -lauseiden väliin:

```
Dir1.Path = Drive1.Drive
```

Tämä lause päivittää hakemistoluettelon Path-ominaisuuden, kun käyttäjä valitsee asemaluettelosta levyaseman. Se kytkee kaksi objektia yhteen siten, että hakemistoluettelossa näkyvät oikean aseman kansiot.

11 Sulje Code-ikkuna napsauttamalla sen oikeassa yläkulmassa olevaa sulkemispainiketta. Kaksoisnapsauta lomakkeen hakemistoluettelo-objektia ja lisää seuraava lause Dir1_Change-aliohjelmaan:

```
File1.Path = Dir1.Path
```

Tämä lause kytkee tiedostoluettelon hakemistoluetteloon siten, että tiedostoluettelossa näkyvät oikean kansion tiedostot.

12 Sulje Code-ikkuna, kaksoisnapsauta lomakkeen tiedostoluettelo-objektia ja lisää seuraavat lauseet File1_Click-aliohjelmaan:

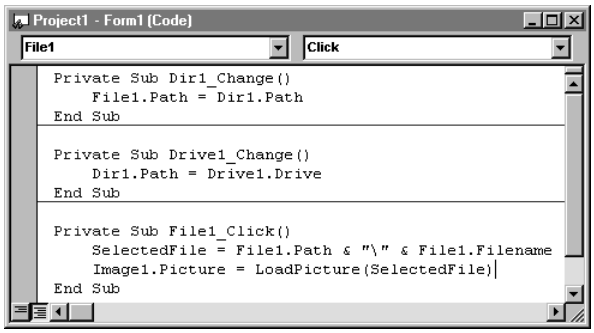
```
SelectedFile = File1.Path & "\" & File1.Filename
Image1.Picture = LoadPicture(SelectedFile)
```

Nämä kaksi lausetta ovat ohjelman ydin. Ensimmäinen lause liittää peräkkäin File1-objektin Path-ominaisuuden, kenoviivan (\) ja File1-objektin Filename-ominaisuuden &-operaattoreilla ja sijoittaa saadun polkunimen SelectedFile-muuttujaan. *Muuttuja* on tilapäinen säilytyspaikka ohjelman tarvitsemille tiedoille. Tässä tapauksessa SelectedFile-muuttuja sisältää käyttäjän valitseman tiedoston täydellisen nimen (myös levyaseman tunnuksen ja kansioiden nimet). Tapahtuma-aliohjelman toinen lause tarvitsee SelectedFile-muuttujaa, kun se näyttää valitun tiedoston lomakkeen kuvaobjektissa (Image1) LoadPicture-funktion ja Picture-ominaisuuden avulla.

Opit lisää operaattoreista, muuttujista ja funktioista harjoituksessa 4.

Kun olet kirjoittanut File1_Click-aliohjelman lauseet, Code-ikkunan pitäisi näyttää seuraavan sivun kuvan mukaiselta. (Kuvassa on suurennettu Code-ikkuna.)

Valmis Browser-ohjelma on kansiossa \Vb5Sbs\Less02.



Nyt olet valmis suorittamaan Browser-ohjelman ja tallentamaan sen levylle.

Browser-ohjelman suorittaminen

Käynnistä-painike

1. Napsauta työkalurivin Käynnistä-painiketta.

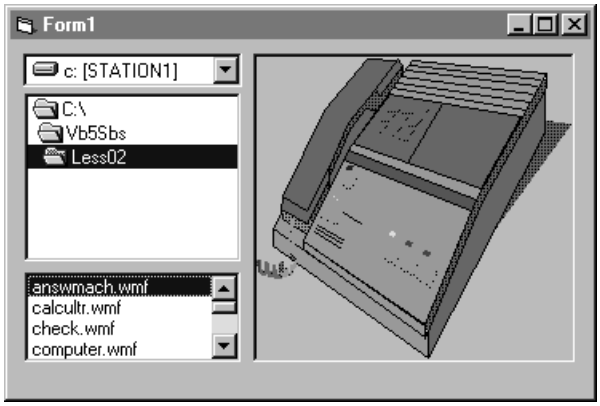
   Browser-ohjelma käynnistyy ohjelmointiympäristön alaisuudessa.

2. Avaa kansio \Vb5Sbs\Less02 käyttäen hakemistoluetteloa.

   Kansion Windows-metatiedostojen nimet näkyvät tiedostoluettelossa.

3. Napsauta tiedostoa answmach.wmf.

   Valitun tiedoston sisältö (vastaajapuhelimen kuva) ilmestyy ohjelman ikkunaan tällaisena:



4. Vieritä luetteloa alaspäin ja napsauta tiedostoa poundbag.wmf.

   Ikkunaan ilmestyy Englannin puntia sisältävän säkin kuva.

5. Katsele järjestelmäsi muita bittikarttoja, Windows-metatiedostoja tai kuvakkeita käyttämällä asema-, hakemisto- ja tiedostoluetteloita. Löydät kansiosta \Windows varmaan useita kiinnostavia bittikarttoja.

   Kun olet kokeillut Browser-ohjelmaa tarpeeksi, lopeta se ja tallenna se levylle.

6 Napsauta lomakkeen sulkemispainiketta.

7 Valitse Tiedosto-valikosta komento Save Project As. Tallenna sekä lomake että projekti nimellä **MyBrowser**.

Entä jos ohjelmani kaatuu?

Jos käytät Browser-ohjelmaa paljon, huomaat ehkä sen aiheuttavan ajonaikaisen virheen eli *kaatuvan* kahdessa tilanteessa. Koska ohjelma on kirjoitettu nopeasti esittelytarkoituksiin, siitä puuttuu lauseet, jotka suojaisivat sen poikkeuksellisilta tilanteilta. Kun ryhdyt kirjoittamaan monimutkaisempia ohjelmia, sinun pitää testata lauseet huolellisesti, jotta ohjelma ei kaadu tavanomaisissa eikä äärimmäisissäkään käyttötilanteissa.

Browser-ohjelman ensimmäinen ongelma on se, että se kaatuu, jos käyttäjä valitsee levyasemaluettelosta aseman, jossa ei ole levyä tai joka ei muusta syystä ole toimintavalmis. Esimerkiksi levykeasemassa voi olla alustamaton levyke tai verkkoasemaan ei ole yhteyttä. Voit todeta tämän tarkistamalla, että levykeasemassa A ei ole levykettä, ja valitsemalla Browser-ohjelman asemaluettelosta vaihtoehdon A:. Ohjelma pysähtyy heti, ja Visual Basic näyttää ilmoituksen "Run-time error '68': Device unavailable." Tämä merkitsee, että Visual Basic ei löytänyt levykettä ja lopetti ohjelman suorituksen, koska se ei pystynyt jatkamaan. Browser-ohjelma luottaa siihen, että käyttäjä ei tee yhtään virhettä levyjen suhteen, mikä ei ehkä ole kovin viisas oletus.

Toinen ongelma liittyy pääkansiossa olevien grafiikkatiedostojen katselemiseen. Koska pääkansiossa sijaitsevan tiedoston polkunimessä on ainoastaan kenoviiva (\), lause

```
SelectedFile = File1.Path & "\" & File1.Filename
```

luo kaksi peräkkäistä kenoviivaa sisältävän polkunimen, joka ei kelpaa. Jos esimerkiksi valitset pääkansiosta tiedoston truck.wmf, polkunimeksi tulee C:\\Truck.wmf. Kun Visual Basic yrittää avata tiedoston, jonka nimessä on kaksi kenoviivaa peräkkäin, se havaitsee ajonaikaisen virheen ja pysäyttää ohjelman.

Nämä ongelmat voidaan välttää kirjoittamalla lauseet niin, että virhettä ei synny (tämä on mahdollista polkunimiongelmassa) tai lisäämällä *virheenkäsittelijöiksi* kutsuttuja erikoisaliohjelmia, joiden avulla ohjelma toipuu virhetilanteesta. Palaamme virheenkäsittelijöihin myöhemmin, mutta voit jo nyt panna merkille, että vaikka Visual Basic selviytyy useimmista tilanteista, käyttäjä voi silloin tällöin aiheuttaa ongelman, joka kaataa hyvinkin suunnitellun ohjelman. Kuvaan virheiden jäljittämistä ja korjaamista harjoituksissa 5 ja 7.

Syöttötietoja keräävät objektit

Visual Basicissa on useita objekteja, joilla kerätään ohjelmassa syöttötietoja. Tekstiruutuun kirjoitetaan tekstiä, valikko sisältää napsautettavia komentoja ja valintaikkunassa on monia osia, jotka voidaan valita yksitellen tai ryhmänä. Seuraavaksi opit käyttämään neljää tärkeää objektia, jotka auttavat keräämään syöttötietoja erilaisissa tilanteissa: valintanappi-, valintaruutu-, luetteloruutu- ja avattavaa luetteloruutuobjektia. Pääset tutustumaan näihin objekteihin ohjelmassa nimeltä Online Shopper. Se toimii käyttöliittymänä Internet-sovellukselle tai muulle ohjelmalle, jolla tilataan tietokoneita ja toimistolaitteita. Kokeile ohjelmaa, niin saat käyttännön kokemusta tiedonsyöttöobjekteista. Sitten kerron, miten näitä objekteja käytetään valikoiden ohella täysimittaisessa ohjelmassa.

Online Shopper -ohjelma

Online Shopper jäljittelee elektronista tilausjärjestelmää, joka näyttää heti, millaisia tuotteita olet tilannut. Jos toimit yrityksessä, jossa tarvitaan paljon tilausten kirjoittamista, voit tulevaisuudessa laajentaa tämän esimerkkiohjelman graafiseksi tilausjärjestelmäksi. (Tällaiset graafiset työkalut ovat suosittuja World Wide Webissä.) Kun kokeilet Online Shopper -ohjelmaa, tarkkaile, miten sen valintanapit, valintaruudut, luetteloruutu ja avattava luetteloruutu toimivat. Ne on luotu Visual Basicissa muutamalla yksinkertaisella toimenpiteellä.

Online Shopper -ohjelman suorittaminen

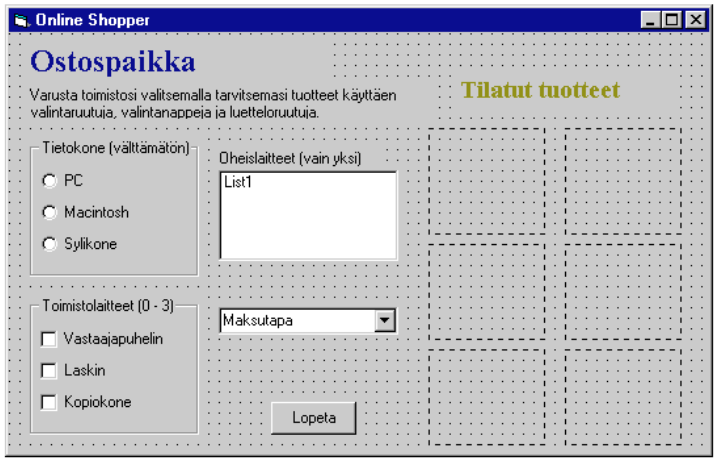
1. Valitse Visual Basicin File-valikosta komento Open Project.

   Open Project -valintaikkuna avautuu.
2. Avaa kansiosta \Vb5Sbs\Less02 tiedosto Online.vbp.
3. Valitse Project-ikkunasta Online-lomake ja napsauta Näytä objekti -painiketta.

Näytä objekti -painike

4. Sulje Properties-, Project-, Form Layout- ja Immediate-ikkunat, jotta näet suuremman osan Online Shopper -lomakkeesta. (Et tarvitse näitä ikkunoita nyt.)

   Online Shopper -lomake tulee näkyviin seuraavan kuvan kaltaisena.



Lomakkeessa on valintanappi-, valintaruutu-, luetteloruutu-, kuva-, komentopainike- ja tekstiobjekteja. Niistä muodostuu pelkistetty tilauksenkirjoitusohjelma, joka havainnollistaa Visual Basicin tiedonsyöttöobjektien toimintaa. Kun Online Shopper -ohjelma käynnistetään, se lataa Windows-metatiedostot aseman C kansiosta \Vb5Sbs\Less02 ja näyttää ne lomakkeen kuudessa kuvakehyksessä.

HUOMAA Jos asensit harjoitustiedostot muualle kuin oletuskansioon C:\Vb5Sbs, metatiedostot lataavien ohjelmalauseiden polkunimet ovat väärät. (Kunkin lauseen alussa on polku c:\vb5sbs\less02, kuten pian näet.) Saat ohjelman toimimaan vaihtamalla harjoitustiedostojen kansion nimeksi \Vb5Sbs tai vaihtamalla polkunimet Code-ikkunassa käyttämällä muokkausnäppäimiä tai Edit-valikon komentoa Replace.

Käynnistäpainike

5 Napsauta työkalurivin Käynnistä-painiketta.

Ohjelma käynnistyy ohjelmointiympäristön alaisuudessa.

Valintanapeilla valitaan luettelosta yksi vaihtoehto.

6 Napsauta Tietokone-ryhmän Sylikone-valintanappia.

Lomakkeen oikeassa reunassa olevalle Tilatut tuotteet -alueelle ilmestyy sylitietokoneen kuva. Tietokone-ryhmässä on joukko *valintanappeja*, joilla käyttäjä antaa syöttötietoja. Käyttäjä voi valita ryhmästä ainoastaan yhden valintanapin kerrallaan. Kun käyttäjä napsauttelee eri valintanappeja, nykyinen valinta ilmenee oikealla olevalta Tilatut tuotteet -alueelta.

7 Napsauta Toimistolaitteet-ryhmän valintaruutuja Vastaajapuhelin, Laskin ja Kopiokone.

Valintaruuduilla valitaan miten monta vaihtoehtoa tahansa.

Valintaruutuja käytetään, kun luettelosta on mahdollista valita useita vaihtoehtoja kerralla. Napsauta Laskin-valintaruutua uudelleen ja huomaa, miten laskimen kuva katoaa lomakkeesta. Koska käyttöliittymän jokainen osa vastaa napsautustapahtumaan heti, lomake on aina valittujen vaihtoehtojen mukainen.

8 Valitse Oheislaitteet-luetteloruudusta Satelliittiantenni.

Luetteloruudussa on vaihteleva määrä vaihtoehtoja, joista valitaan yksi.

Lomakkeeseen ilmestyy satelliittiantennin kuva. *Luetteloruutu* on tapa näyttää luettelo, josta valitaan yksi vaihtoehto. Siinä voi olla hyvin monta vaihtoehtoa (luetteloruudussa on vierityspalkki, jos vaihtoehtoja on enemmän kuin kerralla mahtuu näkyviin), ja valintanapeista poiketen oletusvaihtoehtoa ei tarvita. Visual Basic -ohjelma voi lisätä, poistaa tai lajitella luetteloruudun vaihtoehtoja suorituksen aikana.

9 Valitse avattavasta Maksutapa-luetteloruudusta vaihtoehto Yhdysvaltain dollari (luottokortti ei valitettavasti kelpaa).

Avattava luetteloruutu vie vähemmän tilaa kuin tavallinen luetteloruutu.

Avattava luetteloruutu muistuttaa tavallista luetteloruutua, mutta vie vähemmän tilaa. Visual Basic huolehtii sen avaamisesta, sulkemisesta ja vierittämisestä. Sinun tarvitsee vain kirjoittaa lauseet, jotka lisäävät vaihtoehdot luetteloruutuun ohjelman alussa ja jotka käsittelevät käyttäjän valitseman vaihtoehdon. Näet esimerkkejä näistä tehtävistä Online Shopper -ohjelman lauseissa.

Kun olet tehnyt valintasi, tilauslomakkeen pitäisi näyttää suunnilleen tällaiselta:

10 Harjoittele vielä tekemällä muutamia muutoksia ohjelman tilausluetteloon (kokeile eri tietokoneita, oheislaitteita ja maksutapoja) ja lopeta ohjelma napsauttamalla Lopeta-painiketta.

Kun napsautat Lopeta, ohjelman lomake katoaa, ja ohjelmointiympäristö palaa näkyviin.

Online Shopper -ohjelman tarkasteleminen

Ohjelmalauseita kuvataan tarkemmin harjoituksissa 4, 5 ja 6.

Vaikka sinulla ei ole vielä paljon kokemusta ohjelmalauseista, sinun kannattaa vilkaista muutamia Online Shopper -ohjelman tapahtuma-aliohjelmia nähdäksesi, miten ohjelma käsittelee käyttöliittymän osista saadut syöttötiedot. Näissä aliohjelmissa on If...Then- ja Select Case -lauseita. Kuvaan näitä ja muita ehtorakenteita harjoituksessa 5. Nyt voimme keskittyä Value-ominaisuuteen, joka muuttuu, kun valintaruutu valitaan, ja ListIndex-ominaisuuteen, joka muuttuu, kun luetteloruutu valitaan.

Valintaruutujen ja luetteloruudun lauseet

1. Varmista, että ohjelman suoritus on päättynyt, ja kaksoisnapsauta Toimistolaitteet-ryhmän Vastaajapuhelin-valintaruutua, jotta tapahtuma-aliohjelma ilmestyy Check1_Click Code-ikkunaan.
2. Muuta Code-ikkunan kokoa niin, että näet seuraavat ohjelmalauseet:

Ohjelmarivin lopussa oleva alaviiva (_) tarkoittaa, että lause jatkuu seuraavalla rivillä.

```
Private Sub Check1_Click()
    If Check1.Value = 1 Then
        Image2.Picture = _
          LoadPicture("c:\vb5sbs\less02\answmach.wmf")
        Image2.Visible = True
    Else
        Image2.Visible = False
    End If
End  Sub
```

Check1_Click-aliohjelma sisältää lauseet, jotka suoritetaan käyttäjän napsautettua Vastaajapuhelin-valintaruutua. Tärkeä avainsana on Check1.Value, joka pitää lukea näin: ”ensimmäisen valintaruutuobjektin Value-ominaisuus”. Check1 on ensimmäisen lomakkeeseen lisätyn valintaruudun nimi; seuraavien valintaruutujen nimet ovat Check2, Check3 ja niin edelleen. Value on ominaisuus, joka muuttuu käyttäjän napsauttaessa valintaruutua. Kun valintaruudussa on rasti tai valintamerkki, Value-ominaisuus on 1. Kun valintaruutu on tyhjä, Value-ominaisuus on 0 (nolla).

Value-ominaisuus voidaan määrittää Properties-ikkunassa ohjelmaa kirjoitettaessa, jotta valintaruudulla on oletusasetus. Käyttäjä voi muuttaa ominaisuutta ohjelmaa suoritettaessa napsauttamalla valintaruutua. Edellisessä aliohjelmassa Value-ominaisuuden arvo tutkitaan If...Then...Else-rakenteella. Jos arvo on 1, ohjelma tuo vastaajapuhelimen kuvan lomakkeen toiseen kuvakehykseen ja muuttaa sen näkyväksi. Jos arvo on 0, vastaajapuhelimen kuva on piilossa. Jos tämä tuntuu kummalliselta, älä huolestu. Opit käyttämään ehdollisia rakenteita harjoituksessa 5.

3. Sulje Code-ikkuna ja napsauta lomakkeen Oheislaitteet-luetteloruutua.

Tapahtuma-aliohjelma List1_Click ilmestyy Code-ikkunaan. Siinä on tällaiset lauseet:

```
Private Sub List1_Click()
    Select Case List1.ListIndex
    Case 0
        Image3.Picture = _
          LoadPicture("c:\vb5sbs\less02\harddisk.wmf")
    Case 1
        Image3.Picture = _
          LoadPicture("c:\vb5sbs\less02\printer.wmf")
    Case 2
        Image3.Picture = _
          LoadPicture("c:\vb5sbs\less02\satedish.wmf")
    End Select
    Image3.Visible = True
End Sub
```

Pari sanaa termeistä

Olen jo käyttänyt useita erilaisia termejä kuvatessani Visual Basic -ohjelman osia. En aio määritellä niitä muodollisesti, vaan kuvata ne lyhyesti sekaannusten välttämiseksi. Oletko jo huomannut, mitä eroa on seuraavilla käsitteillä?

Työkalu Työkalu on väline, jolla lisätään Visual Basic -lomakkeeseen objekteja. Valitse työkalu työkaluruudusta ja piirrä objekti lomakkeeseen vetämällä hiirtä. Useimmilla työkaluilla lisätään käyttöliittymän osia, kuten komentopainikkeita, kuvakehyksiä ja luetteloruutuja.

Objekti Objekti on käyttöliittymän osa, joka lisätään Visual Basic -lomakkeeseen työkalulla. Voit siirtää, suurentaa, pienentää ja mukauttaa objekteja ominaisuusasetuksia käyttäen. Objekteilla on myös sisäisiä toimintoja — ne tietävät, miten niiden tulee toimia tietyissä tilanteissa (esimerkiksi luetteloruutu ”osaa” vierittää sisältöään). Voit ohjelmoida Visual Basic -objekteja kirjoittamalla tapahtuma-aliohjelmia, jotka on sovitettu erilaisia tilanteita varten. Visual Basicissa lomake itsekin on objekti.

Ominaisuus Ominaisuus on Visual Basic -objektin arvo tai piirre, esimerkiksi Caption tai ForeColor. Ominaisuudet määritetään suunnittelun aikana Properties-ikkunassa tai suorituksen aikana ohjelmalauseilla. Ominaisuuden arvo määritetään lauseella

Objekti.Ominaisuus = Arvo

jossa *Objekti* on muokattavan objektin nimi, *Ominaisuus* on muutettavan piirteen nimi ja *Arvo* on uusi ominaisuusasetus. Esimerkiksi lauseella

```
Command1.Caption = "Moi"
```

Kun käyttäjä napsauttaa luetteloruudun vaihtoehtoa, Visual Basic palauttaa vaihtoehdon nimen ohjelmaan ominaisuudessa List1.Text.

Tässä ovat käskyt, jotka suoritetaan, kun käyttäjä valitsee vaihtoehdon Oheislaitteet-luetteloruudusta. Tärkeä avainsana on List1.ListIndex, joka luetaan "ensimmäisen luetteloruutuobjektin ListIndex-ominaisuus". Kun käyttäjä on napsauttanut luetteloruudun vaihtoehtoa, ListIndex-ominaisuus palauttaa luvun,joka ilmaisee vaihtoehdon sijainnin luettelossa. Ensimmäisen vaihtoehdon numero on 0, toisen 1 ja niin edelleen.

List1.Text-ominaisuus palauttaa valitun vaihtoehdon varsinaisen tekstin eli nimen. Visual Basic -ohjelmoijat käyttävät usein tätä arvoa ohjelmissaan. Edellinen aliohjelma tutkii List1.ListIndex-ominaisuuden Select Case -rakenteella ja näyttää eri Windows-metatiedoston sen mukaan, millainen arvo ominaisuudella on. Jos arvo on 0, aliohjelma näyttää kiintolevyaseman kuvan. Jos arvo on 2, aliohjelma näyttää satelliittiantennin kuvan. Opit lisää Select Case -rakenteesta harjoituksessa 5.

sijoitetaan ohjelmaa suoritettaessa Command1-objektin Caption-ominaisuudeksi teksti "Moi".

Tapahtuma–aliohjelma Tapahtuma-aliohjelma on ohjelmalauseiden lohko, joka suoritetaan, kun objektiin kohdistuu jokin toimenpide. Kun esimerkiksi ohjelman ensimmäistä komentopainiketta napsautetaan, tapahtuma-aliohjelma Command1_Click käynnistyy. Tapahtuma-aliohjelmat yleensä lukevat ja asettavat ominaisuuksien arvoja ja kutsuvat toisia ohjelmalauseita, jotka suorittavat tarvittavat tehtävät.

Ohjelmalause Ohjelmalause on Visual Basic -avainsanojen yhdistelmä, joka suorittaa tietyn tehtävän. Se voi varata muistitilaa tiedoille, avata tiedoston, suorittaa laskutoimituksia ja hoitaa monia muita tehtäviä.

Muuttuja Muuttuja on tilapäinen säilytyspaikka ohjelman tarvitsemille tiedoille. Ohjelmoija luo muuttujia, kun hän tallentaa laskutoimitusten tuloksia, luo tiedostojen nimiä, käsittelee syöttötietoja ja niin edelleen. Muuttujiin voidaan tallentaa lukuja, nimiä ja ominaisuuksien arvoja.

Metodi Metodi on erikoislause, joka käsittelee tiettyä objektia halutulla tavalla tai suorittaa jonkin muun tehtävän. Ohjelmalauseessa käytetään metodin yhteydessä merkintätapaa

```
Objekti.Metodi Arvo
```

jossa *Objekti* on muokattavan objektin nimi, *Metodi* on käsky, jolla objektia muokataan, ja *Arvo* on valinnainen argumentti metodia varten. Esimerkiksi lause

```
List1.AddItem "Valmis"
```

sijoittaa List1-luetteloruutuun sanan *Valmis* käyttäen AddItem-metodia.

4 Sulje Code-ikkuna ja kaksoisnapsuta lomaketta (ei objektia), jotta itse lomakkeeseen liittyvät ohjelmalauseet tulevat näkyviin.

Form_Load-aliohjelman lauseet suoritetaan ohjelman käynnistyttyä.

Tapahtuma-aliohjelma Form_Load ilmestyy Code-ikkunaan. Se suoritetaan joka kerta, kun Online Shopper -ohjelma käynnistetään eli *ladataan* muistiin. Ohjelmalauseet sijoitetaan tähän erikoisaliohjelmaan, jos ne on suoritettava aina ohjelman käynnistyttyä. Tällaiset lauseet määrittävät usein, myös Online Shopper -ohjelmassa, käyttöliittymän piirteen, jota ei voi luoda ohjelmointiympäristön työkaluilla eikä Properties- ikkunassa. Form_Load-aliohjelma on tällainen:

```
Image1.Picture = LoadPicture("c:\vb5sbs\less02\pcomputr.wmf")
List1.AddItem "Toinen  kiintolevy"
List1.AddItem "Tulostin"
List1.AddItem "Satelliittiantenni"

Combo1.AddItem "Yhdysvaltain  dollari"
Combo1.AddItem "Sekki"
Combo1.AddItem "Englannin  punta"
```

Ensimmäinen lause lataa mikrotietokoneen kuvan sisältävän Windows-metatiedoston ensimmäiseen kuvakehykseen. Tämä on oletusasetus, joka vastaa Tietokone-valintanappia. Kolme seuraavaa lausetta lisäävät vaihtoehdot Oheislaitteet-luetteloruutuun (List1). Lainausmerkein ympäröidyt sanat näkyvät luetteloruudussa. Loput lauseet lisäävät vaihtoehdot avattavaan Maksutapa-luetteloruutuun (Combo1). Tärkeä avainsana on AddItem, joka tarkoittaa luetteloruutuihin liittyvää erikoisfunktiota eli *metodia*.

Metodit ovat erikoislauseita, jotka käsittelevät objektia tai suorittavat jonkin tehtävän sen puolesta, esimerkiksi lisäävät vaihtoehtoja luetteloruutuun. Metodit eroavat ominaisuuksista, jotka sisältävät arvoja, ja tapahtuma-aliohjelmista, jotka suoritetaan käyttäjän käsitellessä objektia. Metodeja voidaan myös jakaa objektien kesken; kun olet oppinut käyttämään yhtä metodia, voit soveltaa sitä erilaisiin tilanteisiin. Kuvaan monia tärkeitä metodeja tämän kirjan harjoitusten yhteydessä.

Online Shopper -ohjelma on nyt käsitelty loppuun. Voit vielä tutkia ohjelman osia, joista olet kiinnostunut, ja siirtyä sitten seuraavaan tehtävään.

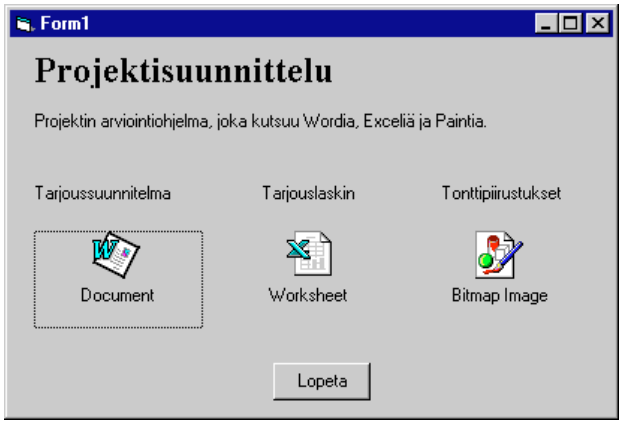
Sovellusten käynnistäminen OLE-objektilla

OLE-objektilla käynnistetään Windows-sovelluksia Visual Basic -ohjelmasta.

Yksi Visual Basicin kiinnostavimmista ominaisuuksista on sen kyky toimia yhdessä muiden Windows-sovellusten kanssa. Sovellus- eli OLE-objektin avulla voit käynnistää sovelluksia Visual Basic -ohjelmasta ja käsitellä monenlaisia tietoja. Voit myös käynnistää muiden sovellusten osia, esimerkiksi Microsoft Wordin oikeinkirjoituksen tarkistimen käyttäen erikoistekniikkaa nimeltä *Automation* (aiemmin OLE Automation). Kuvaan OLE-objekteja ja Automation-tekniikkaa yksityiskohtaisesti harjoituksessa 10. Seuraavassa ohjelmassa näet,

miten OLE-objekti toimii ja miten sillä tehdään (kirjoittamatta yhtään ohjelmalausetta) sovellus nimeltä Bid Estimator. Tämä sovellus avaa Word-, Excel- ja Paint-tiedostoja, jotta käyttäjä voi syöttää rakennusprojektiin liittyviä arvioita, tarjouslaskentatietoja ja tonttipiirustuksia.

Jotta voisit käyttää Bid Estimator -ohjelmaa, kiintolevyllä pitää olla Word, Excel ja Paint. (Paint sisältyy Microsoft Windows 95:een.) Kun luot OLE-objektia, näyttöön tulee Insert Object -valintaikkuna, jossa on lueteltu ohjelmassa käytettävät OLE-objektit. Ellei sinulla ole Wordia, Exceliä tai Paintia, niiden objektit eivät näy Insert Object -valintaikkunassa, mutta voit valita niiden sijasta jonkin muun OLE-objektin. Tämän harjoituksen tarkoitus on kokeilla sovellusten käynnistämistä Visual Basic -ohjelmasta. Valmis ohjelma näyttää suorituksen aikana tällaiselta:



Bid Estimator -ohjelman luominen

1. Valitse File-valikosta New Project ja luo tavanomainen ohjelma napsauttamalla OK.

   Online Shopper -ohjelma katoaa näytöstä, ja ohjelmointiympäristöön ilmestyy uusi lomake.

2. Lisää lomakkeen vasempaan yläkulmaan otsikoksi tekstikehys ”Projektisuunnittelu”. Lisää sen alapuolelle toinen tekstikehys, jossa lukee ”Projektin arviointiohjelma, joka kutsuu Wordia, Exceliä ja Paintia”.

   Varaa ensimmäiselle tekstikehykselle ylimääräistä tilaa, koska sen pistekokoa on tarkoitus suurentaa myöhemmin, ominaisuuksia määritettäessä.

3. Lisää toisen tekstikehyksen alapuolelle vierekkäin ja tasaisin välein (katso yllä olevaa kuvaa) vielä kolme tekstikehystä, joissa lukee ”Tarjoussuunnitelma”, ”Tarjouslaskin” ja ”Tonttipiirustukset”.

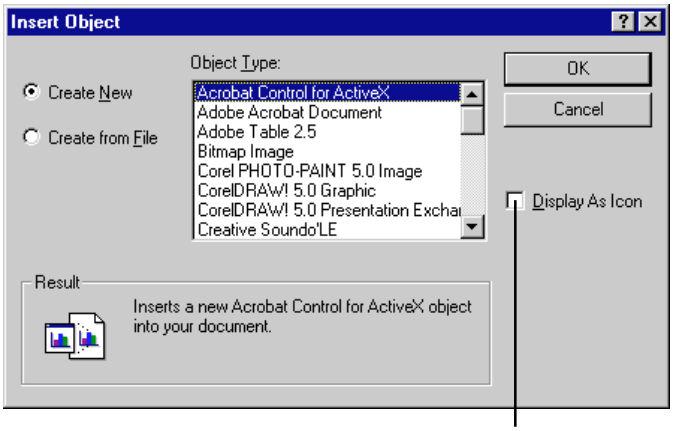
Nämä tekstit toimivat otsikoina OLE-objekteille, joilla käynnistetään Word, Excel ja Paint. Lisää seuraavaksi OLE-objektit.

4 Napsauta työkaluruudun OLE-työkalua.

OLE-työkalu

5 Piirrä Tarjoussuunnitelma-otsikon alapuolelle suunnilleen tulitikkulaatikon kokoinen suorakulmio OLE-objektia varten.

Kun vapautat hiiren painikkeen, Visual Basic avaa alla esitetyn Insert Object -valintaikkunan, jossa on lueteltu käytettävissä olevat sovellus- eli OLE-objektit. Luettelon sisältö riippuu järjestelmästä.



Napsauta tätä, jos haluat näyttää OLE-objektin kuvakkeena

6 Vieritä objektiluetteloa alaspäin ja napsauta Microsoft Word Document -objektia, jos tietokoneessasi on Word for Windows 95.

Ellei sinulla ole Word for Windows 95:tä, valitse Wordin aiempi versio, jokin toinen Windows-tekstinkäsittelyohjelma tai vastaava sovellus.

7 Napsauta Insert Object -valintaikkunan Display As Icon -valintaruutua, jotta sovellus näkyy Visual Basic -ohjelmassa kuvakkeena.

Ellet valitse tätä valintaruutua, OLE-objekti (yleensä asiakirja) näkyy ohjelmassa ikkunana. Käytämme tätä mahdollisuutta vielä monta kertaa tässä kirjassa. Nyt voit kuitenkin valita asetuksen Display As Icon.

8 Napsauta OK, jotta Insert Object -valintaikkuna sulkeutuu ja Word käynnistyy.

Word näyttää tyhjän asiakirjan ikkunassa. Tästä asiakirjasta tulee Bid Estimator -ohjelman *malli*. Siinä pitäisi olla tiedot, joita urakoitsija tarvitsee, esimerkiksi tietoja rakennusyhtiöstä, nimiä, osoitteita, hintoja ja materiaalien kuvauksia.

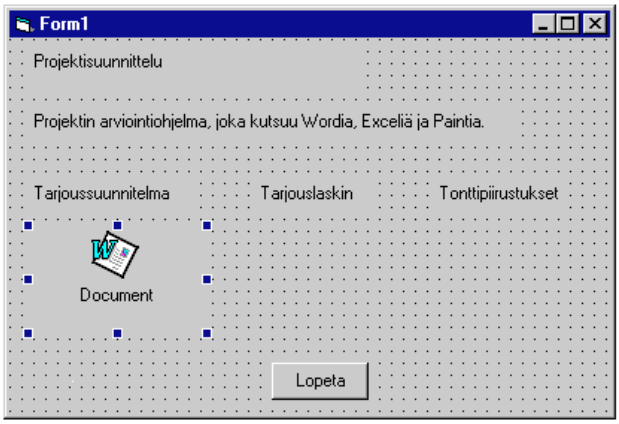
9 Kirjoita malliin teksti **Tietoja tarjouksesta —** ja lisää tämänhetkinen päivämäärä Lisää-valikon (Insert) komennolla Päivämäärä ja aika (Date And Time).

Teksti näkyy Word-asiakirjassa samanlaisena kuin ohjelmaa käytettäessä.

10 Valitse Wordin Tiedosto-valikosta (File) Lopeta (Exit).

Jos Word kysyy, haluatko tallentaa lähdeasiakirjan, vastaa myöntävästi. Tämä kysymys tulee näyttöön, kun suljet tiettyjä sovellusobjekteja.

Kun olet saanut ensimmäisen objektin valmiiksi, lomakkeen pitäisi olla seuraavan kuvan kaltainen. Muuta objektin tai tekstikehyksen kokoa, jos ne peittävät toisensa.



HUOMAA OLE-objektin kolmiulotteinen ulkoasu riippuu Appearance-ominaisuudesta, jolla on arvot 3D ja Flat. Vaiheessa 13 vaihdat jokaisen OLE-objektin Appearance-ominaisuudeksi Flat ja BackColor-ominaisuudeksi vaaleanharmaa, jotta objektit sopivat lomakkeen ulkoasuun.

11 Toista vaiheet 3–10 ja lisää lomakkeeseen Tarjouslaskin-otsikon alapuolelle Microsoft Excel -taulukko-objekti (tai vastaava) ja Tonttipiirustukset-otsikon alle bittikarttakuvaobjekti.

Muista napsauttaa kummallakin kerralla Insert Object -valintaikkunan Display As Icon -valintaruutua. Kirjoita mallitietoja (esimerkiksi huomautuksia tai ohjeita) Excel-taulukkoon ja Paint-piirrokseen, jos osaat käyttää näitä sovelluksia. Järjestelmällinen urakoitsija voisi kirjoittaa

Excel-taulukkoon vaikkapa useita rivejä ja sarakkeita tarjouslaskentatietoja, esimerkiksi puutavarasta, maaleista ja työstä johtuvat kustannukset. Koska voit käynnistää ohjelmastasi muita Windows-sovelluksia, kaikki näiden toiminnot ovat automaattisesti käytettävissäsi — sinun ei tarvitse keksiä pyörää uudelleen!

12 Piirrä lomakkeen alaosaan komentopainike. Kun olet lisännyt painikkeen, kaksoisnapsauta sitä ja kirjoita Command1_Click-aliohjelmaan lause **End**.

End-lause päättää ohjelman, kun käyttäjä on suorittanut haluamansa tehtävät.

13 Määritä objektien ominaisuuksille seuraavat asetukset Properties-ikkunassa:

Objekti	Ominaisuus	Asetus
Command1	Caption	”Lopeta”
Label1	Font	Times New Roman, Bold, 18 pistettä
OLE1	BorderStyle	0 - None
	Appearance	0 - Flat
	BackColor	Light gray
OLE2	BorderStyle	0 - None
	Appearance	0 - Flat
	BackColor	Light gray
OLE3	BorderStyle	0 - None
	Appearance	0 - Flat
	BackColor	Light gray

14 Valitse File-valikosta komento Save Project As ja tallenna lomake levylle nimellä **MyOleBid**. Tallenna myös projekti nimellä **MyOleBid**.

Kun olet valmis, MyOleBid-lomakkeen pitäisi olla seuraavan sivun kuvan mukainen.

Valmis OleBid-ohjelma on kansiossa \Vb5Sbs\Less02.

Käynnistä ohjelma ja kokeile, miten OLE-objektit toimivat suorituksen aikana.

Uuden Bid Estimator -ohjelman suorittaminen

1. Napsauta työkalurivin Käynnistä-painiketta.

   Ohjelma käynnistyy ohjelmointiympäristön alaisuudessa. OLE1-objekti (asiakirjakuvake) on ympäröity pisteviivalla osoitukseksi siitä, että se on aktiivinen.

> **HUOMAA** Pisteviiva on tärkeä, jos ohjelmaa käytetään näppäimillä. Kun käyttäjä painaa ENTER, ohjelma valitsee aktiivisen eli pisteviivalla ympäröidyn objektin. Käyttäjä voi aktivoida jonkin muun objektin painamalla SARKAINNÄPPÄINTÄ tai napsauttamalla objektia. Objektien selausjärjestystä muutetaan vaihtamalla kunkin objektin TabIndex-ominaisuus.

2. Kaksoisnapsauta ohjelman asiakirjakuvaketta.

   Word-tekstinkäsittelyohjelma käynnistyy ja näyttää Word-malliasiakirjan ikkunassa.

3. Kirjoita muutama rivi tekstiä (kuvittele olevasi projektin tärkeä alihankkija) ja palaa MyOleBid-ohjelmaan valitsemalla Wordin Tiedosto-valikosta (File) komento Lopeta (Exit).

4. Kaksoisnapsauta ohjelman taulukkokuvaketta.

   Excel-taulukkolaskentaohjelma käynnistyy ja näyttää Excel-mallitaulukon ikkunassa.

5. Kirjoita taulukkoon muutaman rivin ja sarakkeen verran tietoja (voit vapaasti kokeilla Excelin funktioita ja muotoilukomentoja). Palaa MyOleBid-ohjelmaan valitsemalla Tiedosto-valikosta (File) komento Lopeta (Exit).
6. Kaksoisnapsauta bittikarttakuvaketta.

   Paint-maalausohjelma käynnistyy ja näyttää tyhjän piirustusikkunan. Paint on yksinkertainen ohjelma, jossa on piirtotyökaluja ja värivalikoima alkeellisten piirrosten tekemistä varten.
7. Laadi Paintilla jonkinlainen tonttipiirustus (tai ainakin kuvittele niin) ja valitse Tiedosto-valikosta (File) komento Lopeta ja palaa (Exit & Return To).
8. Lopeta ohjelma napsauttamalla Lopeta-painiketta.

Siinä kaikki! Olet tehnyt ensimmäisen ohjelmasi, joka käyttää Microsoft Office -sovellusobjekteja. Tällä menetelmällä voit sisällyttää ohjelmaan mitä tahansa järjestelmään asennettua sovellusta vastaavan objektin. Kokeile seuraavaksi toisenlaista Windows-ympäristön tiedostoa — tietokantaa, jossa on asiakkaiden nimiä ja osoitteita.

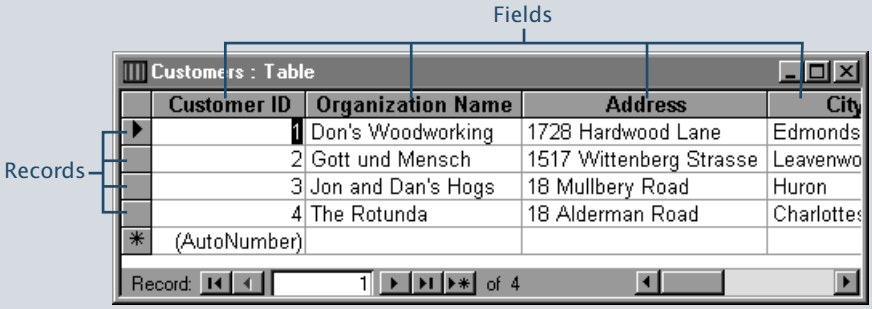
Microsoft Access -tietokannan katseleminen dataobjektin avulla

Jos työskentelet yrityksessä tai jaat säännöllisesti tietoja muiden käyttäjien kanssa, tarvitset ehkä tietokantoja, joihin tallennetaan tärkeitä tietoja asiakkaista, työntekijöistä tai keskeneräisistä projekteista. *Tietokanta* on elektronisesti tallennettujen tietojen järjestetty kokoelma. Tietokantasovellukset, kuten Microsoft Access, dBase ja Paradox, ovat erikoisohjelmia, jotka muodostavat ja käsittelevät tietokantoihin tallennettavia tietoja. Niissä on välineet, joilla luodaan tietokanta, muokataan sen sisältöä ja etsitään tietoja. Visual Basicissa on tietokantojen käsittelemisen helpottamiseksi kolme objektia, joilla näytetään ja muokataan tietokantatiedoston sisältöä. Tärkein näistä on dataobjekti, jolla käsitellään tietokannan kenttiä ja tietueita suoraan lomakkeessa. Kokeile dataobjektin käyttämistä nyt tuomalla näkyviin Access-tietokannan Shops.mdb sisältö.

HUOMAA Tässä tehtävässä tarvittava Access-tiedosto on valmiina kansiossa Less02, joten voit suorittaa seuraavat vaiheet, vaikka sinulla ei olisikaan Microsoft Accessia. Voit myös valita Access-tiedoston sijasta jonkin oman tietokantatiedostosi.

Kentät ja tietueet

Tietokannan rakennetta kuvataan käyttäen kahta tärkeää termiä: *kenttä* ja *tietue*. Kentät ovat tietokantaan tallennettujen tietojen luokkia. Esimerkiksi asiakastietokannan tavanomaisia kenttiä ovat asiakkaan nimi, osoite, puhelinnumero ja lisätiedot. Tiettyä asiakasta tai yritystä kuvaavien tietojen kokoelma on nimeltään tietue. Henkilöstötietokannassa tietue sisältää yhden työntekijän tiedot. Kun tietokanta luodaan, tiedot kirjoitetaan kentistä ja tietueista muodostuvaan tauluun. Tietueet vastaavat usein taulun rivejä ja kentät sarakkeita.



Dataobjektin luominen

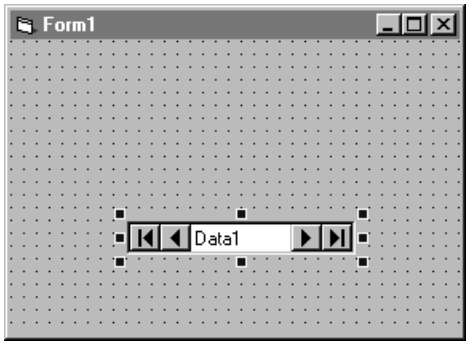
1. Valitse File-valikosta komento New Project ja luo tavallinen ohjelma napsauttamalla OK.

   MyOleBid-ohjelma katoaa näytöstä, ja ohjelmointiympäristöön ilmestyy uusi lomake. Tallenna MyOleBid-ohjelmaan tekemäsi muutokset, jos Visual Basic kysyy, tallennetaanko ne.

Data-työkalu

2. Napsauta työkaluruudun Data-työkalua.
3. Siirrä hiiren osoitin keskelle lomakkeen alaosaa ja piirrä suorakulmio.

   Lomakkeeseen ilmestyy dataobjekti nimeltä Data1, kuten seuraavan sivun kuvasta näkyy.



Objekti sisältää nuolipainikkeet, joilla selataan tietokannan tietueita ohjelman suorituksen aikana. Objektissa on myös teksti (tällä hetkellä Data1), joka kuvaa objektin avulla käsiteltävää tietokantaa. Yleensä teksti on sama kuin tietokannan nimi. Objektin ääripäissä olevilla painikkeilla siirrytään tietokannan alkuun tai loppuun.

Voit käynnistää Visual Basicilla monia edistyksellisiä tietokantatoimintoja. Tässä tehtävässä on tarkoitus näyttää Shops.mdb-tietokannan Name-kenttä. Sitten voit selata tietokantaa ja tutkia siihen tallennettuja nimiä. Jotta voisit näyttää Name-kentän lomakkeessa, tarvitset sitä varten uuden objektin. Koska esitettävä tieto on tekstiä, lisää lomakkeeseen tekstiruutuobjekti. Sijoita sen viereen myös tekstiobjekti, joka ilmaisee tekstiruudun merkityksen. Muodosta lopuksi yhteys dataobjektin ja tekstiruutuobjektin välille eli *sido* ne toisiinsa käyttämällä useita ominaisuusasetuksia.

Teksti- ja tekstiruutuobjektien luominen

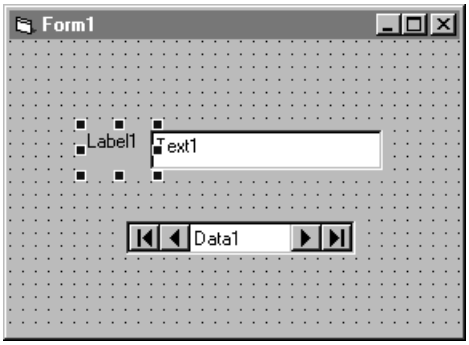
Tekstiruutu-työkalu

Teksti-työkalu

1. Napsauta työkaluruudun Tekstiruutu-työkalua.
2. Piirrä lomakkeeseen tekstiruutu dataobjektin yläpuolelle.

   Tekstiruudun pitäisi olla suunnilleen samankokoinen kuin dataobjekti. Siinä tulee olla tarpeeksi tilaa 20 merkkiä pitkälle liikkeen nimelle.
3. Napsauta työkalurivin Teksti-työkalua.
4. Lisää tekstikehys tekstiruudun vasemmalle puolelle.

Kun olet lisännyt objektit, lomakkeen pitäisi näyttää seuraavan kuvan mukaiselta.



Määritä seuraavaksi objektien ominaisuudet.

Objektien ominaisuuksien määrittäminen

1. Napsauta dataobjektia ja sitten työkalurivin Properties-ikkuna-painiketta.
2. Tarkista Properties-ikkunasta, että Connect-ominaisuus on Access (oletusasetus).

   Connect-ominaisuus ilmaisee käytettävän tietokantamuodon. Visual Basic pystyy lukemaan Access-, dBase-, FoxPro- ja Paradox-tietokantoja.
3. Määritä Properties-ikkunassa DatabaseName-ominaisuudeksi c:\vb5sbs\less02\shops.mdb valitsemalla tämä tiedosto DatabaseName-valintaikkunasta.

   Shops.mdb on Access-mallitietokanta, joka sinun pitää avata tällä ohjelmalla. Se sisältää nimiä, osoitteita, puhelinnumeroita ja muita tietoja, jotka koskevat kuvitteellisia liikkeitä.
4. Valitse Properties-ikkunasta RecordSource-ominaisuus ja napsauta avattavan luetteloruudun nuolipainiketta. Kun tietokantataulujen luettelo tulee näkyviin, valitse siitä Customers.

   RecordSource-ominaisuudella määritetään taulu (tietokokoelma), joka halutaan avata tietokannasta.
5. Määritä Properties-ikkunassa dataobjektin Caption-ominaisuudeksi ”Shops.mdb”.

   Dataobjektiin ilmestyy teksti Shops.mdb, joka ilmaisee tietokannan nimen käyttäjälle. Muuta nyt tekstiruudun DataSource-ominaisuutta, jotta tekstiruudun ja dataobjektin välille muodostuu yhteys.
6. Napsauta tekstiruutuobjektia ja työkalurivin Properties-ikkuna-painiketta.
7. Valitse Properties-ikkunasta DataSource-ominaisuus, napsauta avattavan luetteloruudun nuolipainiketta ja valitse luettelosta Data1 (ensimmäinen dataobjekti).

8. Valitse Properties-ikkunasta DataField-ominaisuus, napsauta avattavan luetteloruudun nuolipainiketta ja valitse luettelosta OrganizationName (näytettävän kentän nimi).
9. Napsauta tekstiobjektia ja työkalurivin Properties-ikkuna-painiketta ja vaihda Caption-ominaisuudeksi "Name".

   Tämä tekstiobjekti ilmaisee tietokantakentän merkityksen, kun ohjelma on käynnissä. Lomakkeeseen kannattaa aina lisätä selittäviä tekstejä, varsinkin jos käsitellään tietokannan kenttiä.
10. Tallenna sekä lomake että projekti nimellä **MyData**.

Siinä kaikki! Nyt voit käynnistää ohjelman.

MyData-ohjelman suorittaminen

Valmis Data-ohjelma on kansiossa \Vb5Sbs\Less02.

1. Napsauta työkalurivin Käynnistä-painiketta.

   Ohjelma käynnistyy ohjelmointiympäristön alaisuudessa seuraavan kuvan kaltaisena:

   Visual Basic lukee Shops.mdb-tietokannan ja näyttää ensimmäisen OrganizationName-kentän sisällön tekstiruudussa. Tarkastele muiden tietueiden sisältöjä napsauttamalla dataobjektin painikkeita.
2. Napsauta dataobjektin sisempää oikealle osoittavaa nuolta.

   Toinen tietokantaan tallennettu nimi ilmestyy tekstiruutuun.
3. Napsauta dataobjektin ulompaa oikealle osoittavaa nuolta.

   Viimeinen tietokantaan tallennettu nimi ilmestyy tekstiruutuun.
4. Napsauta dataobjektin ulompaa vasemmalle osoittavaa nuolta.

   Ensimmäinen tietokantaan tallennettu nimi ilmestyy tekstiruutuun.
5. Lopeta ohjelma napsauttamalla lomakkeen sulkemispainiketta.

Tietokannan muokkaaminen

Tietokantaobjektilla voidaan myös muuttaa tietokannan sisältöä. Jos haluat vaihtaa Shops.mdb-tietokantaan tallennetun nimen, käynnistä MyData-ohjelma ja tuo vaihdettava nimi näkyviin. Napsauta tekstiruutua ja tee nimeen haluamasi muutokset. Kun siirryt toiseen tietueeseen, muutos kopioituu heti alkuperäiseen tietokantaan. Kokeile tätä seuraavaksi.

Tietokannassa olevan nimen vaihtaminen

1. Käynnistä MyData-ohjelma napsauttamalla työkalurivin Käynnistä-painiketta.

   Ensimmäinen tietokantaan tallennettu nimi näkyy tekstiruudussa.
2. Korosta nimi hiirellä, paina DEL ja kirjoita **Penan kuljetus**.
3. Siirry seuraavaan tietueeseen napsauttamalla dataobjektin sisempää oikealle osoittavaa nuolta.

   Ensimmäiseen tietueeseen tallentuu nimi Penan kuljetus.
4. Palaa ensimmäiseen tietueeseen napsauttamalla dataobjektin sisempää vasemmalle osoittavaa nuolta.

   Näet nimen Penan kuljetus.
5. Lopeta ohjelma napsauttamalla lomakkeen sulkemispainiketta.

Kuten huomaat, dataobjekti on helppo tapa käsitellä olemassa olevia tietokantoja. Voit tuoda minkä tahansa kentän näkyviin ja muuttaa sen sisältöä. Opit lisää dataobjekteista ja tiedostojen käsittelemisestä harjoituksessa 11.

Lisätietoja: ActiveX-objektien asentaminen

Voit laajentaa Visual Basicia asentamalla ActiveX-objekteja, jotka kuuluvat Visual Basiciin tai jotka olet hankkinut joltakin muulta valmistajalta. ActiveX-objektit sijoitetaan työkaluruutuun projektia varten Project-valikon komennolla Components. Uudet ActiveX-objektit perustuvat 32-bittiseen ActiveX-tekniikkaan, joka on Microsoft-standardi sovellusohjelmien, käyttöjärjestelmien ja Internet-apuvälineiden ohjelmoitaville objekteille. ActiveX-tiedostojen tarkenne on .ocx, ja niillä lisätään uusia toimintoja Windows 95- ja Windows NT -ohjelmiin. Aiempien, Visual Basicin 16-bittisille versioille ja Windows 3.1:lle suunniteltujen objektien eli kontrollien tarkenne on .vbx. Sinun pitäisi käyttää uusia .ocx-objekteja, jos mahdollista. Visual Basic osaa monissa tapauksissa korvata vanhat objektit uusilla vastineillaan, kun uudet objektit on asennettu.

ActiveX-objektien Grid ja CommonDialog asentaminen

Visual Basic 5 Learning Edition sisältää ActiveX-objekteja, joita voit käyttää omissa projekteissasi. Jos teet esimerkiksi ohjelmaa, joka näyttää tietoja taulukkomuodossa, asenna Grid-objekti, jolla muodostetaan lomakkeeseen soluruudukko. Grid-objekti näyttää lähes samanlaiselta kuin Excel-taulukko. Tämä ActiveX-objekti sijaitsee tiedostossa Grid32.ocx. Toinen hyödyllinen ActiveX-objekti on CommonDialog, jolla tehdään tavanomaisia valintaikkunoita, kuten Avaa (Open) ja Tallenna nimellä (Save As). Se sijaitsee tiedostossa Comdlg32.ocx.

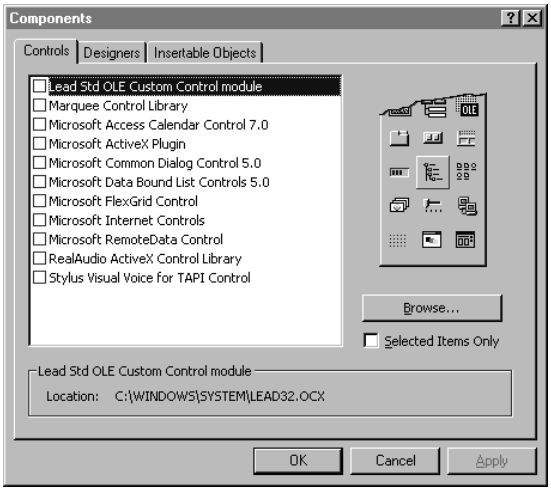
Asenna ActiveX-objektit näin:

1. Valitse File-valikosta komento New Project ja luo tavanomainen ohjelmatiedosto napsauttamalla OK.

   Tallenna MyData-ohjelmaan tekemäsi muutokset tarvittaessa.

2. Valitse Project-valikosta komento Components ja napsauta Controls-välilehteä.

   Components-valintaikkuna avautuu seuraavan kuvan näköisenä:



Controls-välilehdessä on lueteltu aakkosjärjestyksessä järjestelmän ActiveX-objektit, jotka voidaan lisätä projektin työkaluruutuun. Jotta voisit kehittää ohjelmia joustavasti, jokaisella projektilla on oma työkaluruutunsa, joka sisältää Visual Basicin oletusobjektit ja valitsemasi ActiveX-objektit. Siksi jokainen tähän projektiin lisäämäsi objekti näkyy vain sen työkaluruudussa. Seuraavissa vaiheissa lisäät työkaluruutuun Grid-objektin (Grid32.ocx) ja CommonDialog-objektin (Comdlg32.ocx).

VIHJE Components-valintaikkunassa on myös Insertable Objects -välilehti, jolla lisätään sovellusobjekteja projektin työkaluruutuun. *Lisättävä objekti* on kokonaisuus, joka on peräisin toisesta Windows-sovelluksesta, esimerkiksi Word-asiakirja tai Excel-taulukko. Tällaiset objektit ovat yhtä hyödyllisiä kuin ActiveX-objektit.

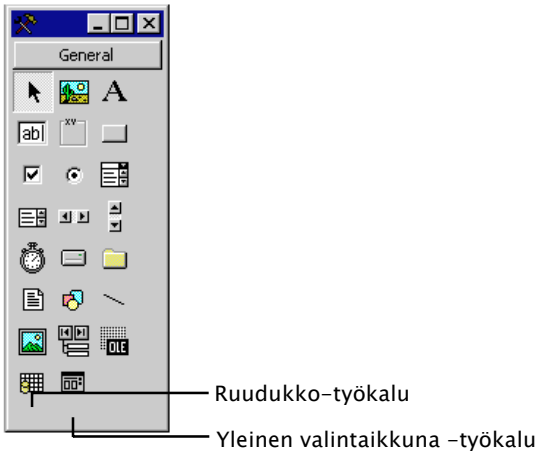
3. Napsauta valintaruutua, joka on objektin Microsoft Common Dialog Control 5.0 vieressä.

   Visual Basic valitsee ActiveX-objektin ja näyttää sitä vastaavan .ocx-tiedoston sijainnin valintaikkunan alareunassa.

4. Napsauta valintaruutua, joka on objektin Microsoft Grid Control vieressä.

5. Lisää valitut ActiveX-objektit tämän projektin työkaluruutuun napsauttamalla OK.

   Työkaluruudussa näkyy kaksi uutta objektia, jotka on esitetty seuraavassa kuvassa:



Grid- ja CommonDialog-objekti toimivat täsmälleen samoin kuin työkaluruudun muut Visual Basic -objektit. Ellet tietäisi niiden olevan ActiveX-objekteja, sinun olisi vaikea erottaa niitä muista objekteista. Valitse ActiveX-objekti napsauttamalla ja lisää se lomakkeeseen samalla tavoin kuin mikä tahansa objekti. ActiveX-objekteillakin on muutettavat ominaisuusasetukset, ja niitä voidaan ohjata ohjelmalauseilla aivan kuin muitakin tässä harjoituksessa esitettyjä objekteja.

ActiveX-objektit ja Windows 95

Jos teet ohjelmia, joiden on tarkoitus toimia Windows 95:ssä tai Windows NT 4.0:ssa, sinun on hyvä tietää, että Visual Basic 5 Professional Edition ja Enterprise Edition sisältävät erikoiskokoelman ActiveX-objekteja, jotka on suunniteltu varsinkin Windows 95-yhteensopivia käyttöliittymiä varten. Seuraavassa taulukossa on esitelty lyhyesti nämä ActiveX-objektit, joista useimmat aktivoituvat valitsemalla Components-valintaikkunasta vaihtoehto Microsoft Windows Common Controls 5.0. Jos aiot käyttää näitä kehittyneitä toimintoja ohjelmissasi, sinun kannattaa hankkia Visual Basic Professional tai Enterprise Edition ohjelmointiajan säästämiseksi.

ActiveX-objekti	Kuvaus
RichTextBox	Parannettu TextBox-objekti, jossa on enemmän muotoilutoimintoja.
ListView	Näyttää tiedosto- ja kuvakeluettelot samalla tavoin kuin Windows Resurssienhallinta (Explorer).
TreeView	Näyttää hierarkkiset tiedot (esimerkiksi kansiot ja alikansiot) samalla tavoin kuin Windows Resurssienhallinta.
Tabstrip	Luo valintaikkunoita, joissa on Windows 95 -tyyliset välilehdet.
Toolbar	Lisää ohjelmiin vakiotyökalurivejä.
Slider	Lisää ohjelmaan liukusäätimen, jolla käyttäjä voi nopeasti siirtyä toiseen paikkaan tiedostossa tai mediaesityksessä.
ProgressBar	Ilmoittaa käyttäjälle, kuinka kauan tehtävän suorittaminen kestää.
StatusBar	Näyttää tilatietoja vakiomuodossa.
ImageList	Sisältää kokoelman kuvia, joita muut objektit voivat käyttää.
UpDown	Näyttää ylä- ja alanuolet, joilla voidaan suurentaa ja pienentää arvoja.
Animation	Näyttää videoleikkeen (AVI).

Jos haluat jatkaa seuraavaan harjoitukseen

➤ Jätä Visual Basic käyntiin ja siirry harjoitukseen 3.

Jos Visual Basic kysyy, tallennetaanko muutokset nykyiseen projektiin, napsauta No.

Jos haluat lopettaa Visual Basicin

➤ Valitse File-valikosta komento Exit.

Jos näyttöön tulee Save-valintaikkuna, napsauta No. Sinun ei tarvitse tallentaa tätä projektia ja sen ActiveX-objektien luetteloa.

Harjoituksen tiivistelmä

Tehtävä	Ohje	Painike
Luoda tekstiruutu	Napsauta Tekstiruutu-työkalua ja piirrä ruutu.	
Luoda komento-painike	Napsauta Komentopainike-työkalua ja piirrä painike.	
Muuttaa ominaisuuttа suorituksen aikana	Muuta ominaisuuden arvoa ohjelma-lauseella. Esimerkki: `Text1.Text = "Terve!"`	
Luoda levyasema-luetteloruutu	Napsauta Levyasemaluettelo-työkalua ja piirrä luetteloruutu.	
Luoda hakemisto-luetteloruutu	Napsauta Hakemistoluettelo-työkalua ja piirrä luetteloruutu.	
Luoda tiedosto-luetteloruutu	Napsauta Tiedostoluettelo-työkalua ja piirrä luetteloruutu.	
Estää ohjelman kaatuminen	Kirjoita ohjelmalauseista muodostuva tapahtuma-aliohjelma (katso harjoitusta 7).	

Lukea kuvatiedosto suorituksen aikana	Kutsu LoadPicture-funktiota ja sijoita tulos kuva- tai kuvakehysobjektin Picture-ominaisuuteen. Lauseen muoto on tällainen:
	`Objekti``.Picture = LoadPicture(`*`Tiedosto`*`)`
	Objekti on objektin nimi, ja *Tiedosto* on muuttuja, joka sisältää kuvatiedoston nimen. Esimerkki:
	`Tiedosto = "c:\veturi.bmp"`
	`Image1.Picture = LoadPicture(Tiedosto)`	

Luoda valintanappi	Käytä Valintanappi-työkalua. Kun lisäät useita valintanappeja, sijoita ne Kehys-työkalulla piirtämäsi suorakulmion sisään.	
Luoda valintaruutu	Napsauta Valintaruutu-työkalua ja piirrä valintaruutu.	

Tehtävä	Ohje	Painike
Luoda luetteloruutu	Napsauta Luetteloruutu-työkalua ja piirrä luetteloruutu.	
Luoda avattava luetteloruutu	Napsauta Avattava luetteloruutu -työkalua ja piirrä luetteloruutu.	
Lisätä vaihtoehtoja luetteloruutuun	Lisää AddItem-metodin sisältäviä lauseita ohjelman Form_Load-aliohjelmaan. Esimerkki: `List1.AddItem "Tulostin"`	
Käynnistää Windows-sovelluksia	Piirrä lomakkeeseen suorakulmio sovellusta varten OLE-työkalulla ja valitse sopiva sovellusobjekti Insert Object -valintaikkunasta.	
Näyttää tietokannan sisältö	Luo Data-työkalulla objekti, jolla selataan tietokantaa, ja sido tämä dataobjekti toiseen objektiin, joka pystyy näyttämään tietokannan tietueet (tavallisesti tekstiruutuobjektiin).	
Muokata tietokannan tietueita	Tuo tietue näkyviin ohjelmalla. Muokkaa tietuetta tekstiruudussa ja napsauta dataobjektin nuolta, jotta muutos tallentuu levylle.	
Asentaa ActiveX-objekteja	Valitse Project-valikosta komento Components ja napsauta Controls-välilehteä. Valitse projektin työkaluruutuun lisättävät ActiveX-objektit ja napsauta OK-painiketta.	

Kun tarvitset tietoja aiheesta	Toimi näin
Visual Basicin objektit	Valitse Visual Basic Help-valikosta Microsoft Visual Basic Help Topics, valitse Index-välilehti, kirjoita objektin nimi ja napsauta Display.

Tietoja seuraavasta harjoituksesta

Seuraavassa harjoituksessa opit lisäämään ohjelmiisi valikoita ja käsittelemään käyttäjän antamia tietoja valintaikkunoissa. Näet myös, miten CommonDialog-objektilla muodostetaan hyödyllisiä valintaikkunoita tietojen syöttämistä varten.

Harjoitus

3

Valikoiden ja valintaikkunoiden käyttö

Arvioitu aika
45 min.

Tässä harjoituksessa opit

- lisäämään ohjelmiin valikoita Menu Editorilla
- käsittelemään valikkoja ohjelmalauseilla
- näyttämään vakiovalintaikkunoita valintaikkunaobjekteilla.

Harjoituksessa 2 käytit useita Visual Basic -objekteja, joilla syötetään tietoja ohjelmaan. Tässä harjoituksessa opit näyttämään käyttäjälle vaihtoehtoja huoliteltujen valikoiden ja valintaikkunoiden avulla. Valikko sijaitsee valikkorivillä ja sisältää toisiinsa liittyvien komentojen luettelon. Kun napsautat valikon otsikkoa, ohjelma näyttää valikon komentojen luettelon. Useimmat valikkokomennot suoritetaan heti, kun ne valitaan. Jos käyttäjä esimerkiksi valitsee Muokkaa-valikosta (Edit) komennon Kopioi (Copy), valitut tiedot kopioituvat Windowsin Leikepöydälle. Jos komennon nimen jäljessä on kolme pistettä (...), Visual Basic näyttää valintaikkunan lisätietojen antamista varten ennen kuin komento suoritetaan. Tässä harjoituksessa opit käyttämään Menu Editoria ja CommonDialog-objektia lisätessäsi ohjelmiin valikoita ja vakiovalintaikkunoita.

Valikoiden lisääminen Menu Editorilla

Menu Editor on graafinen väline, jolla käsitellään ohjelmien valikoita. Sillä tehdään uusia valikoita, muokataan ja järjestellään entisiä ja poistetaan vanhoja. Sillä myös lisätään valikoihin erikoistoimintoja, kuten valintakirjaimia,

pikanäppäimiä ja valintamerkkejä. Kun olet lisännyt lomakkeeseen valikoita, voit käsitellä valikkokomennot tapahtuma-aliohjelmilla. Seuraavaksi käytät Menu Editoria luodessasi Kello-valikon, jossa on komennot nykyisen päivämäärän ja ajan näyttämistä varten.

Valikon luominen

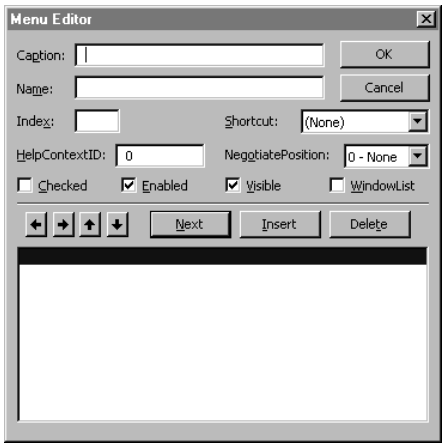
1. Käynnistä Visual Basic.

   Jos ohjelmointiympäristö on jo käynnissä, luo tavallinen ohjelmatiedosto valitsemalla File-valikosta komento New Project.

Menu Editor -painike

2. Napsauta työkalurivin Menu Editor -painiketta.

   Menu Editor käynnistyy seuraavan kuvan näköisenä:



Menu Editor helpottaa valikoiden luomista ja muokkaamista.

Menu Editor -valintaikkunassa on komentoja ja asetuksia, joilla muodostetaan valikoita. Caption-tekstiruutuun pitää kirjoittaa valikon otsikko, joka näkyy valikkorivillä, ja Name-tekstiruutuun valikon nimi, johon viitataan ohjelmassa. Nämä ovat valikon tärkeimmät asetukset. Muut asetukset, kuten Index, HelpContextID, Shortcut ja Checked, ovat valinnaisia. Shortcut-asetusta kuvataan tämän luvun lopussa.

Kun napsautat Next-painiketta, Menu Editor -valintaikkuna tyhjenee seuraavan valikon lisäämistä varten. Valintaikkunan alaosassa on luetteloruutu, jossa näkyvät lisäämäsi valikkokomennot ja valikon yleinen rakenne. Luo seuraavaksi Kello-valikko Menu Editorilla.

3. Kirjoita Caption-tekstiruutuun **Kello** ja paina SARKAINNÄPPÄINTÄ.

   Sana *Kello* on ensimmäisen valikon otsikko, ja kohdistin siirtyy Name-tekstiruutuun. Kirjoittamasi otsikko näkyy myös valintaikkunan alareunan valikkoluetteloruudussa.

Valikon nimeen on tapana lisätä etuliite mnu.

4 Kirjoita Name-tekstiruutuun **mnuClock**.

mnuClock on valikon nimi ohjelmassa. Valikoiden nimet on tapana aloittaa ohjelmalauseissa etuliitteellä *mnu*. Kun merkitset käyttöliittymän osien nimet kolmikirjaimisilla etuliitteillä, pystyt helpommin erottamaan tapahtuma-aliohjelmat toisistaan ohjelmiesi kasvaessa. Etuliitteet auttavat myös löytämään käyttöliittymän osat Code-ikkunasta.

HUOMAA Harjoituksen 8 kohdassa ”Lisätietoja” luetellaan Visual Basic -objektien nimeämissääntöjä.

5 Lisää Kello-valikko ohjelmaan napsauttamalla Next-painiketta.

Kello-valikko näkyy valikkorivillä, ja Menu Editorin valintaikkuna tyhjenee seuraavaa valikkokomentoa varten. Valikon otsikko näkyy edelleen valintaikkunan alaosan luetteloruudussa. Kun muodostat valikkoja, jokainen valikko ja komento ilmestyy luetteloruutuun, jotta näet valikkojärjestelmän rakenteen.

6 Kirjoita Caption-tekstiruutuun **Päivä**, paina SARKAINNÄPPÄINTÄ ja kirjoita **mnuDateItem**.

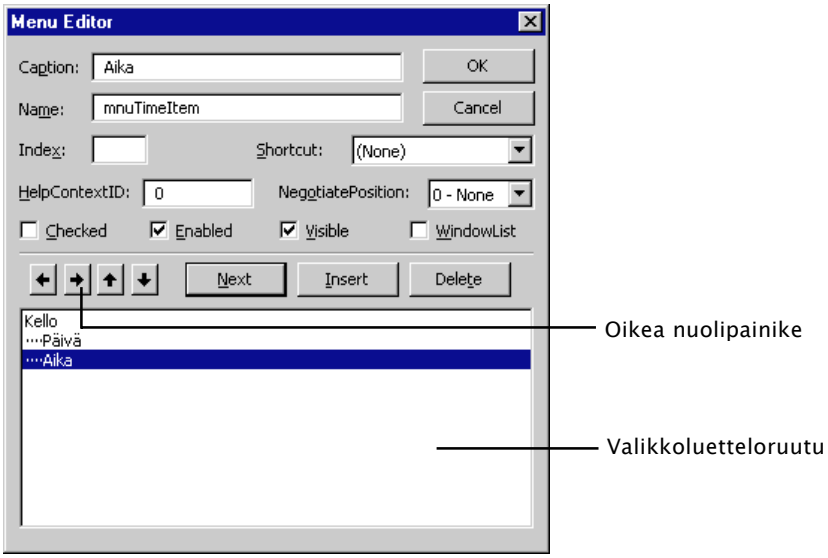
Teksti Päivä ilmestyy luetteloruutuun. Koska Päivä on valikkokomento eikä valikko, sen nimeämisessä käytettiin toista merkintätapaa: Kirjoitit Name-tekstiruutuun komennon jälkeen sanan *Item*. Tämä jälkiliite selventää komentojen ja valikoiden eroa Code-ikkunassa.

7 Kun Päivä-teksti on korostettu luetteloruudussa, napsauta Menu Editorin oikeaa nuolipainiketta.

Päivä-teksti siirtyy yhden sisennyksen (neljän välilyönnin) verran oikealle osoitukseksi siitä, että se on valikkokomento. Tekstin sijainti luetteloruudussa määrää, onko kyseessä valikon otsikko (alkaa vasemmasta reunasta), komento (yksi sisennys), alivalikon otsikko (kaksi sisennystä) vai alivalikon komento (kolme sisennystä). Siirrä tekstiä vasemmalle tai oikealle napsauttamalla Menu Editor -valintaikkunan vasenta tai oikeaa nuolipainiketta. Lisää Kello-valikkoon komento nimeltä Aika.

8 Napsauta Next-painiketta, kirjoita **Aika**, paina SARKAINNÄPPÄINTÄ ja kirjoita **mnuTimeItem**.

Aika-komento ilmestyy luetteloruutuun tällaisena:



Huomaa, että Menu Editor oletti tekstin tarkoittavan valikkokomentoa ja sisensi sitä yhden tason verran. Olet nyt lisännyt molemmat komennot Kello-valikkoon, ja voit lopettaa Menu Editorin.

9 Sulje Menu Editor -valintaikkuna napsauttamalla OK.

Kun valintaikkuna on sulkeutunut, näet lomakkeen, jossa on valikkorivi ja Kello-valikko. Avaa Kello-valikko, jotta näet sen sisällön.

10 Napsauta Kello-valikkoa.

Päivä- ja Aika-komennot tulevat näkyviin.

Kun napsautat valikkokomentoa ohjelmointiympäristössä, näyttöön tulee tapahtuma-aliohjelma, joka suoritetaan kyseisen komennon valitsemisen jälkeen. Määritä Päivä- ja Aika-komentojen tapahtuma-aliohjelmat hieman myöhemmin. Lisää ensin valikoihin näppäintoimintoja.

11 Sulje Kello-valikko napsauttamalla lomaketta tai painamalla kahdesti ESC.

Valintakirjainten lisääminen valikkokomentoihin

Voit määrittää valintakirjaimen lisäämällä sen eteen et-merkin (&).

Visual Basicilla on helppo määrittää näppäimet, jotka edustavat valikoita ja komentoja. Valintakirjain tarkoittaa näppäintä, jolla komento valitaan. Kun käyttäjä avaa valikon, valintakirjaimet näkyvät alleviivattuina komentojen nimissä. Kun haluat lisätä valikkokomentoon valintakirjaimen, käynnistä Menu Editor ja lisää komennon nimeen et-merkki (&) valintakirjaimen edelle. Tämän jälkeen valintakirjain on käytettävissä.

Valikoiden muodostamissääntöjä

Windows-sovelluksen jokaisen valikon ja valikkokomennon nimi aloitetaan isolla kirjaimella. Valikkorivin ensimmäiset valikot ovat useimmiten Tiedosto (File) ja Muokkaa (Edit), ja viimeinen valikko on Ohje (Help). Muita yleisiä valikoiden nimiä ovat Näytä (View), Muotoile (Format) ja Ikkuna (Window). Yritä käyttää selkeitä ja johdonmukaisia nimiä. Valikoiden ja komentojen pitäisi olla helppoja valita ja muistuttaa muissa Windows-sovelluksissa käytettyjä mahdollisimman tarkoin. Kun laadit valikoita, noudata seuraavia ohjeita:

- Valitse lyhyt ja selkeä nimi, jossa on mieluiten yksi tai kaksi sanaa.
- Määritä jokaiselle komennolle valintakirjain. Käytä alkukirjainta, jos mahdollista.
- Jos komento on kaksiasentoinen kytkin, lisää sen edelle valintamerkki, kun se on aktiivinen. Valintamerkki lisätään napsauttamalla Menu Editorin Checked-valintaruutua tai asettamalla komennon Checked-ominaisuudeksi True.
- Lisää kolme pistettä (...) sellaisen komennon jälkeen, joka avaa valintaikkunan lisätietojen antamista varten.
- Käytä vakiintuneita merkintätapoja, kuten etuliitettä *mnu* ja jälkiliitettä *Item*, kun nimeät valikkokomentoja.

Lisää nyt Kello-valikkoon valintakirjaimet.

Valintakirjainten lisääminen

Menu Editor -painike

1. Napsauta työkalurivin Menu Editor -painiketta.

   Menu Editor käynnistyy, ja valikkoluetteloruudussa näkyvät ohjelman valikot ja valikkokomennot. Valintaikkunassa on valmiina Kello-valikon otsikko ja nimi.

2. Napsauta sanan *Kello* vasenta puolta Caption-tekstiruudussa.

   Kohdistin vilkkuu *Kello*-sanan K-kirjaimen vasemmalla puolella.

3. Määritä Kello-valikon valintakirjaimeksi K kirjoittamalla &.

   Tekstiruutuun ilmestyy et-merkki.

4. Napsauta luetteloruudun Päivä-komentoa.

   Päivä-komennon otsikko ja nimi näkyvät valintaikkunassa.

5. Lisää Caption-tekstiruutuun P-kirjaimen edelle et-merkki.

   Päivä-komennon valintakirjain on nyt P.

6. Napsauta luetteloruudun Aika-komentoa.

   Aika-komennon otsikko ja nimi näkyvät valintaikkunassa.

7 Lisää Caption-tekstiruutuun A-kirjaimen edelle et-merkki.

Aika-komennon valintakirjain on nyt A.

8 Lopeta Menu Editor napsauttamalla OK.

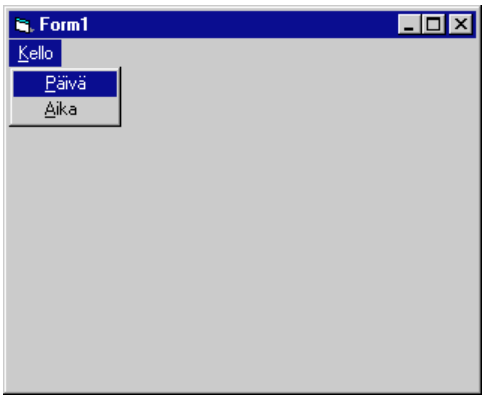
Käynnistä tekemäsi ohjelma, jotta näet, millaisilta valintakirjaimet näyttävät Kello-valikossa.

Käynnistä-painike

9 Napsauta Käynnistä-painiketta.

10 Napsauta Kello-valikkoa.

Kello-valikon otsikossa ja komennoissa on alleviivatut valintakirjaimet.



11 Lopeta ohjelma napsauttamalla Lopeta-painiketta.

Lopeta-painike

Harjoittele Menu Editorin käyttöä vaihtamalla Kello-valikon Päivä- ja Aika-komentojen järjestys. Komentojen järjestyksen muuttaminen on hyvä osata, koska voit hyvinkin huomata, että alkuaan määrittämäsi järjestys ei ole paras mahdollinen.

Valikkokomentojen järjestyksen muuttaminen

1 Napsauta työkalurivin Menu Editor -painiketta.

Menu Editor käynnistyy.

2 Valitse luetteloruudusta Aika-komento.

Time-komennon otsikko ja nimi ilmestyvät valintaikkunaan.

3 Napsauta ylänuolipainiketta.

Aika-komento siirtyy Päivä-komennon yläpuolelle.

4 Napsauta OK-painiketta.

Menu Editor -valintaikkuna sulkeutuu, ja Kello-valikon Päivä- ja Aika-komennot ovat vaihtaneet paikkaa. Voit napsauttaa myös Menu Editorin alanuolipainiketta, joka siirtää valittua komentoa alaspäin luetteloruudussa.

Olet saanut Kello-valikon ulkoasun valmiiksi. Määritä seuraavaksi tapahtuma-aliohjelmat, jotka suorittavat käyttäjän valitsemat komennot.

HUOMAA Voit myös lisätä uusia valikoita ja komentoja ja poistaa tarpeettomia Menu Editorilla. Napsauta valikkoluetteloruudun kohtaa, jonka edelle haluat lisätä uuden vaihtoehdon, ja napsauta Insert-painiketta. Menu Editor lisää luetteloon tyhjän rivin, ja voit määrittää sen sisällön Caption- ja Name-tekstiruuduissa. Kun haluat poistaa vaihtoehdon luettelosta, napsauta sitä ja Delete-painiketta.

Valikkokomentojen käsittely

Valikkokomennon käsittelee komentoon kytketty tapahtuma-aliohjelma.

Kun valikot ja komennot on lisätty valikkoriville, ne muuttuvat ohjelman objekteiksi. Jotta ne tekisivät jotakin, sinun pitää kirjoittaa niitä varten tapahtuma-aliohjelmat. Valikkoaliohjelmat sisältävät yleensä ohjelmalauseita, jotka näyttävät tai käsittelevät lomakkeen tietoja ja muuttavat yhtä tai useita valikko-ominaisuuksia. Jos käyttäjän pitää antaa lisätietoja komentoa varten, tapahtuma-aliohjelma avaa usein valintaikkunan, joka perustuu yleiseen valintaikkunaobjektiin tai tiedonsyöttöobjektiin.

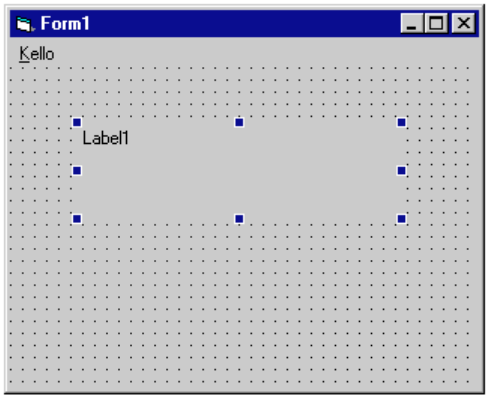
Seuraavaksi sinun pitää lisätä lomakkeeseen tekstiobjekti, joka näyttää Kello-valikon Aika- ja Päivä-komentojen tulokset.

Tekstiobjektin lisääminen lomakkeeseen

Teksti-työkalu

1. Napsauta työkalupaletin Teksti-työkalua.
2. Piirrä lomakkeen keskelle pieni tekstiobjekti.

   Tekstiobjekti ilmestyy lomakkeeseen. Sen nimi ohjelmassa on Label1. Lomakkeen pitäisi näyttää tällaiselta:



Määritä Label1-objektin ominaisuudet Properties-ikkunassa.

3 Määritä tekstiobjektin ominaisuudet näin:

Objekti	Ominaisuus	Asetus
Label1	Alignment	2 - Center
	Border Style	1 - Fixed Single
	Caption	(Tyhjä)
	Font	MS Sans Serif, Bold, 14 pist.

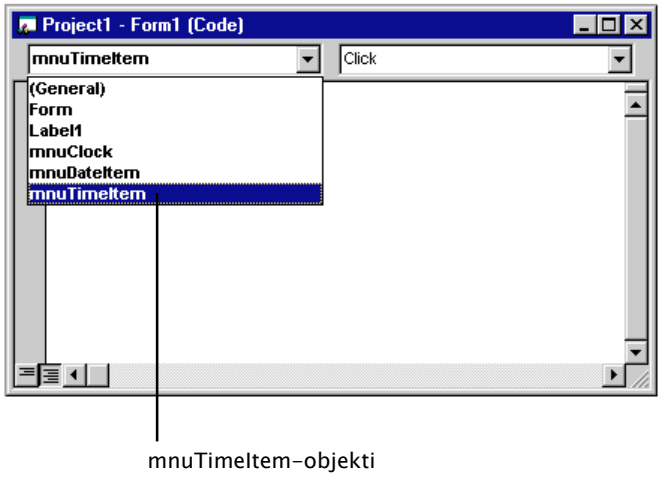
Lisää Time- ja Date-tapahtuma-aliohjelmiin lauseet, jotka käsittelevät valikkokomennot.

HUOMAA Seuraavissa tehtävissä sinun pitää kirjoittaa ohjelmalauseet, jotka suoritetaan käyttäjän valittua komennon. Kirjoita lauseet täsmälleen niin kuin ne on esitetty. En kerro vielä, miten lauseet toimivat, vaan miten niistä muodostetaan toimiva käyttöliittymä. Perehdyt ohjelmalauseiden toimintaan harjoituksissa 4–6.

Valikkoaliohjelmien muokkaaminen

1 Napsauta Project-ikkunan Näytä lauseet -painiketta, jotta Code-ikkuna avautuu.

2 Avaa objektiluettelo ja valitse siitä mnuTimeItem.



Tapahtuma-aliohjelma mnuTimeItem_Click ilmestyy Code-ikkunaan. mnuTimeItem on ohjelman sisäinen nimi, jonka annoit Aika-komennolle Menu Editorissa. Kun käyttäjä valitsee Aika-komennon, mnuTime-Item_Click-aliohjelma käynnistyy.

3 Paina VÄLINÄPPÄINTÄ neljästi ja kirjoita

```
Label1.Caption = Time
```

Tämä lause näyttää tietokoneen kellosta luetun ajan Label1-objektissa ja korvaa objektin entisen tekstin. Käytä Time-funktiota, kun haluat näyttää sekunnintarkan ajan ohjelmassasi.

HUOMAA Visual Basicin Time-funktio palauttaa järjestelmän kellonajan. Tämä asetetaan Ohjauspaneelin (Control Panel) toiminnolla Päivämäärä/Aika (Date/Time). Ajan esitysmuoto valitaan Ohjauspaneelin toiminnolla Maa-asetukset (Regional Settings).

4 Paina ALANUOLINÄPPÄINTÄ.

Visual Basic tutkii rivin ja muuttaa tarvittaessa välejä ja isoja ja pieniä kirjaimia. Visual Basic tarkistaa, onko ohjelmarivillä kielioppivirheitä, kun syötät rivin painamalla ENTER, YLÄNUOLI tai ALANUOLI.

5 Valitse objektiluettelosta mnuDateItem.

Tapahtuma-aliohjelma mnuDateItem_Click näkyy Code-ikkunassa. Se suoritetaan, kun käyttäjä valitsee Clock-valikosta komennon Päivä.

6 Paina VÄLINÄPPÄINTÄ neljästi ja kirjoita

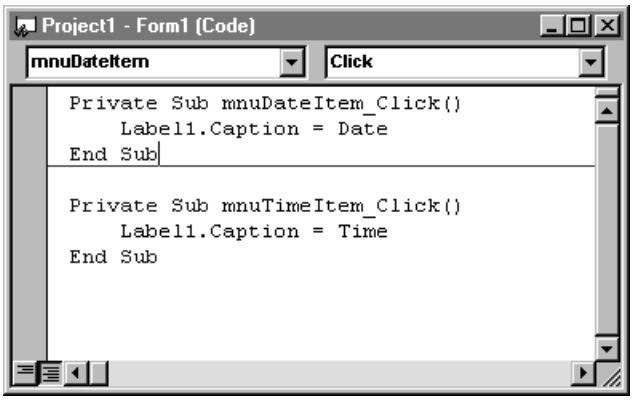
```
Label1.Caption = Date
```

Tämä lause näyttää tietokoneen kellosta luetun päivämäärän Label1-objektissa ja korvaa objektin entisen tekstin. Käytä Date-funktiota, kun haluat näyttää nykyisen päivämäärän ohjelmassasi.

HUOMAA Visual Basicin Date-funktio palauttaa järjestelmän päivämäärän. Tämä asetetaan Ohjauspaneelin (Control Panel) toiminnolla Päivämäärä/Aika (Date/Time). Päivämäärän esitysmuoto valitaan Ohjauspaneelin toiminnolla Maa-asetukset (Regional Settings).

7 Syötä rivi painamalla ALANUOLINÄPPÄINTÄ.

Project-ikkunan pitäisi näyttää tällaiselta:



8 Sulje Code-ikkuna.

Nyt olet saanut valikko-ohjelman valmiiksi. Tallenna lomake ja projekti nimellä MyMenu.

MyMenu-ohjelman tallentaminen

Tallenna projekti –painike

1 Napsauta työkalurivin Tallenna projekti -painiketta.

Tämä painike on vaihtoehto File-valikon komennolle Save Project.

2 Valitse kansio \Vb5Sbs\Less03, kirjoita projektilomakkeen nimeksi **MyMenu** ja paina ENTER.

Lomake tallentuu levylle nimellä MyMenu.frm, ja Save Project As -valintaikkuna avautuu.

3 Kirjoita projektin nimeksi **MyMenu** ja paina ENTER.

Projekti tallentuu levylle nimellä MyMenu.vbp.

Ohjelma on nyt valmis suoritettavaksi.

MyMenu-ohjelman suorittaminen

Käynnistä–painike

1 Napsauta työkalurivin Käynnistä-painiketta.

MyMenu-ohjelma käynnistyy ohjelmointiympäristön alaisuudessa.

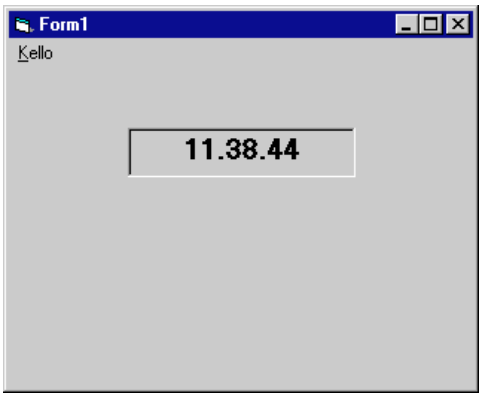
2 Napsauta valikkorivin Kello-valikkoa.

Kello-valikon sisältö tulee näkyviin.

Valmis Menu-ohjelma on kansiossa \Vb5Sbs\Less03.

3 Napsauta Aika-komentoa.

Järjestelmäkellosta luettu aika ilmestyy valintaikkunaan seuraavan kuvan mukaisena:



Tuo valintaikkunaan tämänhetkinen päivämäärä käyttämällä määrittämiäsi valintanäppäimiä.

4 Paina ALT-näppäintä.

Valikkorivin Kello-valikko muuttuu korostetuksi.

5 Avaa Kello-valikko painamalla K.

Kello-valikon komennot tulevat näkyviin.

6 Näytä tämänhetkinen päivämäärä painamalla P.

Päivämäärä tulee näkyviin tekstiobjektin sisäpuolelle.

Lopeta-painike

7 Lopeta ohjelma napsauttamalla Lopeta-painiketta.

Onnittelut! Olet nyt tehnyt toimivan ohjelman, jota ohjataan käyttäen valikoita ja valintanäppäimiä. Seuraavaksi opit avaamaan vakiovalintaikkunoita valikoiden kautta.

Kellofunktiot

Käytettävissä on kymmenen funktiota, jotka lukevat päivämääriä ja aikoja järjestelmäkellosta. Niillä voit lisätä ohjelmiin kalentereita, kelloja ja hälytyksiä. Seuraavassa taulukossa on lueteltu kaikki kellofunktiot. Löydät lisätietoja Visual Basicin ohjeista.

Funktio	Kuvaus
Time	Palauttaa järjestelmäkellosta luetun ajan.
Date	Palauttaa järjestelmäkellosta luetun päivämäärän.
Now	Palauttaa koodatun arvon, joka ilmaisee nykyisen päivämäärän ja ajan. Tätä funktiota käytetään yleensä muiden kellofunktioiden argumenttina.
Hour (*aika*)	Palauttaa annetun ajan tuntiosan (0–23).
Minute (*aika*)	Palauttaa annetun ajan minuuttiosan (0–59).
Second (*aika*)	Palauttaa annetun ajan sekuntiosan (0–59).
Day (*päivä*)	Palauttaa nykyisen päivän numeron (1–31).
Month (*päivä*)	Palauttaa kuluvan kuukauden numeron (1–12).
Year (*päivä*)	Palauttaa annetun päivämäärän vuosiosan.
Weekday (*päivä*)	Palauttaa annettua päivämäärää vastaavan viikonpäivän numeron (1 on sunnuntai, 2 maanantai ja niin edelleen).

Yleisten valintaikkunaobjektien käyttö

Yleistä valintaikkunaobjektia käyttäen voidaan lisätä ohjelmaan viisi erilaista vakiovalintaikkunaa. Kukin vakiovalintaikkuna saadaan samasta yleisestä objektista. Avaa valintaikkuna lisäämällä ohjelmaan yleinen valintaikkunametodi. (Kuten edellä mainittiin, metodi on komento, joka suorittaa objektiin liittyvän tehtävän.) Yleisen valintaikkunan sisältö määritetään asettamalla sopivat ominaisuudet. Kun käyttäjä täydentää yleisen valintaikkunan ohjelmaa suoritettaessa, tulokset tallentuvat valintaikkunaobjektin ominaisuuksiin, ja ohjelma voi lukea nämä ja sitten päättää, mitä sen tulee tehdä.

Seuraavan sivun taulukossa on lueteltu viisi yleistä valintaikkunaa, jotka voidaan johtaa yleisestä valintaikkunaobjektista käyttämällä metodeja.

Valintaikkuna	Kuvaus	Metodi
Avaa (Open)	Lukee olemassa olevan tiedoston nimen, levyaseman ja kansion.	ShowOpen
Tallenna nimellä (Save As)	Lukee uuden tiedoston nimen, levyaseman ja kansion.	ShowSave
Tulosta (Print)	Määrittää tulostusasetuksia.	ShowPrinter
Fontti (Font)	Valitsee uuden fontin ja tekstityylin.	ShowFont
Väri (Color)	Valitsee värin valikoimasta.	ShowColor

Tallenna sekä lomake että projekti uudella nimellä, kun vaihdat projektin nimen. Tallenna aina lomake ensin.

Seuraavissa tehtävissä sinun pitää lisätä MyMenu-ohjelmaan uusi valikko ja kokeilla Open- ja Color-valintaikkunoiden käyttämistä. Säilytä alkuperäinen MyMenu-ohjelma tallentamalla MyMenu-lomake ja -projekti nimellä MyDialog ennen kuin alat tehdä muutoksia.

MyMenu-tiedostojen tallentaminen nimellä MyDialog

1. Ellei MyMenu.vbp-tiedosto ole avoinna, lue se levyltä valitsemalla File-valikosta komento Open Project.

   Ellet kirjoittanut MyMenu-ohjelmaa, avaa kansiosta \Vb5Sbs\Less03 tiedosto MyMenu.vbp, jossa on samat ohjelmalauseet.

2. Valitse File-valikosta komento Save MyMenu.frm As.

   Save File As -valintaikkuna avautuu.

3. Valitse kansio \Vb5Sbs\Less03, kirjoita **MyDialog.frm** ja paina ENTER.

   MyMenu-lomakkeen kopio tallentuu nimellä MyDialog.frm.

VAROITUS Ellet ensin tallenna lomaketta uudella nimellä, MyMenu- ja MyDialog-ohjelmat käyttävät samaa lomaketta.

4. Valitse File-valikosta komento Save Project As.

   Save Project As -valintaikkuna avautuu.

5. Valitse kansio \Vb5Sbs\Less03, kirjoita **MyDialog.vbp** ja paina ENTER.

   MyMenu-projektitiedoston kopio tallentuu nimellä MyDialog.vbp.

Yleisen valintaikkunaobjektin lisääminen

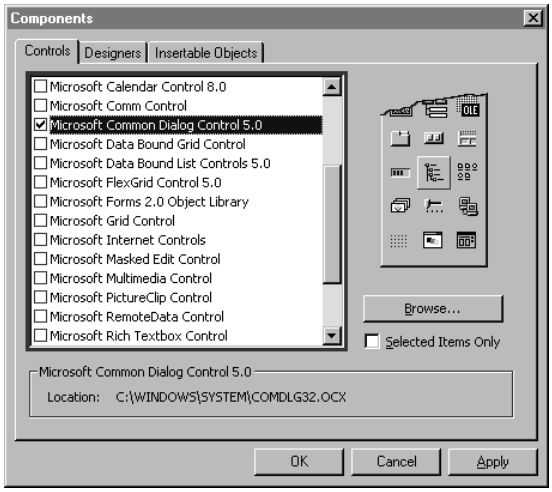
Yleinen valinta-ikkunaobjekti ei näy käyttäjälle suorituksen aikana.

Lisää lomakkeeseen yleinen valintaikkunaobjekti käyttämällä Yleinen valinta-ikkuna -työkalua. Yleinen valintaikkunaobjekti ei näy käyttäjälle ohjelman suorituksen aikana, ja sen kokoa ei voi muuttaa. Koska objekti on näkymätön, se voidaan sijoittaa lomakkeessa mihin tahansa. Kun olet lisännyt tällaisen objektin lomakkeeseen, voit käyttää ohjelmassa mitä tahansa viidestä yleisestä valinta-ikkunasta.

CommonDialog-työkalun lisääminen työkalupalettiin

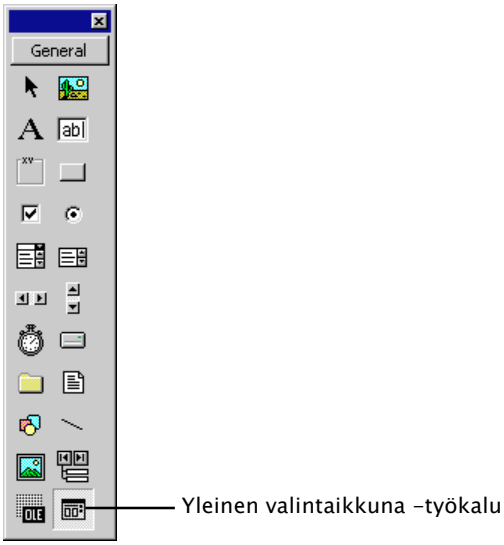
Ellei työkalupaletissa ole Yleinen valintaikkuna -työkalua, lisää se Project-valikon komennolla Components. Toimi näin:

1. Valitse Project-valikosta komento Components.
2. Napsauta Controls-välilehteä ja valitse Microsoft Common Dialog Control 5.0.



3. Napsauta OK.

   Työkalupaletissa on nyt Yleinen valintaikkuna -työkalu, joka on esitetty seuraavan sivun kuvassa.



Yleisen valintaikkunaobjektin lisääminen

Yleinen valintaikkuna -työkalu

1. Napsauta työkalurivin Yleinen valintaikkuna -työkalua.
2. Sijoita yleinen valintaikkunaobjekti lomakkeen vasempaan alakulmaan.

   Kun olet sijoittanut objektin paikalleen, sen koko muuttuu automaattisesti. Yleinen valintaikkunaobjekti on nyt valmis käytettäväksi ohjelmassa.

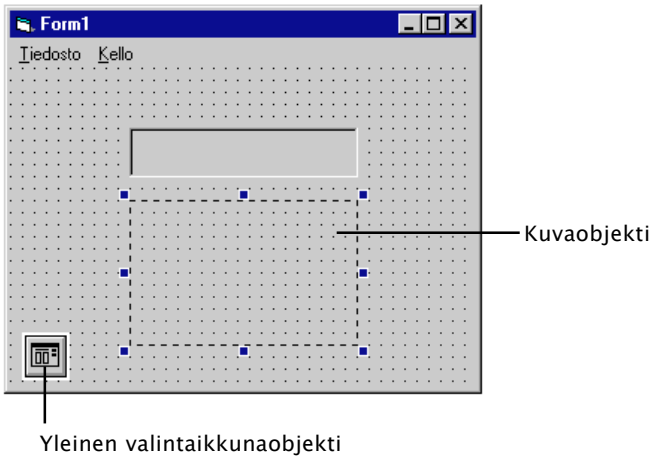
Lisää kuvaobjekti käyttämällä Kuva-työkalua. Se näyttää grafiikkatiedoston, jonka käyttäjä valitsee ohjelman Open-valintaikkunasta.

Kuvaobjektin lisääminen

Kuva-työkalu

1. Napsauta työkalupaletin Kuva-työkalua.
2. Lisää lomakkeeseen kuvaobjekti tekstiobjektin alapuolelle.
3. Määritä Properties-ikkunassa kuvaobjektin Stretch-ominaisuudeksi True.

   Kun olet lopettanut, lomakkeen tulisi olla seuraavalla sivulla esitetyn kuvan näköinen.



Lisää MyDialog-ohjelmaan Tiedosto-valikko Menu Editorilla.

Tiedosto-valikon lisääminen

Menu Editor -painike

1. Valitse lomakeobjekti napsauttamalla lomaketta.

   Lomake täytyy valita ennen valikoiden lisäämistä tai muokkaamista.

2. Napsauta työkalurivin Menu Editor -painiketta.

   Menu Editor käynnistyy ja näyttää MyDialog-ohjelman nykyisen valikkorakenteen. Lisää ohjelmaan Tiedosto-valikko, jossa on komennot Avaa, Sulje ja Lopeta.

3. Napsauta Insert-painiketta neljästi.

   Valikkoluetteloruudun alussa on nyt neljä tyhjää riviä Tiedosto-valikolle ja sen komennoille.

4. Napsauta Caption-tekstiruutua, kirjoita **&Tiedosto**, paina SARKAIN-NÄPPÄINTÄ, kirjoita **mnuFile** ja napsauta Next-painiketta.

   Luetteloruudussa on nyt Tiedosto-valikko, jonka valintakirjain on T.

5. Kirjoita **&Avaa...**, paina SARKAINNÄPPÄINTÄ ja kirjoita **mnuOpenItem**. Napsauta oikeaa nuolipainiketta ja Next-painiketta.

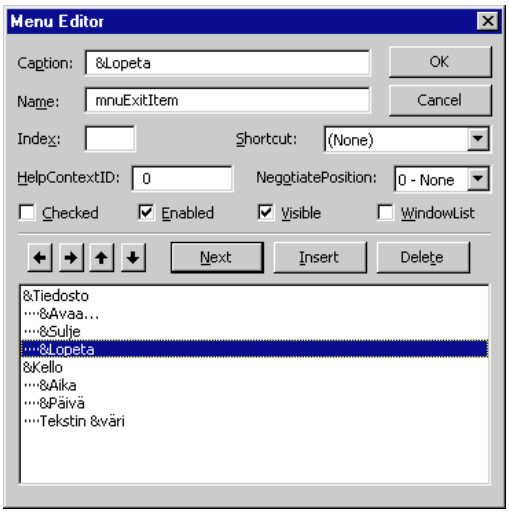
   Luetteloruutuun ilmestyy Avaa-komento, jolla on tarkoitus avata Windows -metatiedostoja ja joka on sisennetty yhden tason verran. Koska komennolla avataan valintaikkuna, komennon nimi päättyy kolmeen pisteeseen.

6. Kirjoita **&Sulje**, paina SARKAINNÄPPÄINTÄ, kirjoita **mnuCloseItem**, napsauta oikeaa nuolipainiketta ja lopuksi Next-painiketta.

   Luetteloruutuun ilmestyy Sulje-komento, jolla suljetaan avoin tiedosto.

7 Kirjoita **&Lopeta**, paina SARKAINNÄPPÄINTÄ, kirjoita **mnuExitItem** ja napsauta oikeaa nuolipainiketta.

Luetteloruutuun ilmestyy Lopeta-komento, jolla lopetetaan MyDialog-ohjelma. Menu Editorin pitäisi näyttää tällaiselta:



Valikkokomennon poistaminen käytöstä

Windows-sovelluksissa on tavallista, että kaikki valikkokomennot eivät ole käytettävissä samanaikaisesti. Esimerkiksi Muokkaa-valikon (Edit) komento Liitä (Paste) voidaan valita vain, jos Leikepöydällä on tietoja. Valikkokomento poistetaan käytöstä kumoamalla sitä vastaavan Enabled-valintaruudun valinta Menu Editorissa. Kun komento ei ole käytettävissä, se näkyy himmennettynä.

Seuraavaksi sinun pitää poistaa käytöstä Sulje-komento, jota on tarkoitus käyttää vain silloin, kun ohjelmalla on avattu tiedosto. Myöhemmin tässä harjoituksessa lisäät Open-aliohjelmaan lauseen, joka ottaa Sulje-komennon käyttöön sopivalla hetkellä.

Sulje-komennon poistaminen käytöstä

1 Valitse luetteloruudusta Sulje-komento.

Komennon otsikko ja nimi ilmestyvät valintaikkunaan.

2 Napsauta Enabled-valintaruutua, jotta siitä häviää rasti.

Kun rastia ei ole, komentoa ei voi valita.

Lisää Kello-valikkoon komento Tekstin väri, joka havainnollistaa yleisen Color-valintaikkunan käyttämistä. Color-valintaikkuna palauttaa väriasetuksen ohjelmaan CommonDialog1.Color-ominaisuuden kautta. Vaihda tämän ominaisuuden avulla ohjelman tekstiobjektin väri.

Tekstin väri -komennon lisääminen Kello-valikkoon

1. Valitse luetteloruudun viimeinen komento eli Päivä.

   TextColor-komento on tarkoitus lisätä Kello-valikon loppuun.

2. Napsauta Next-painiketta.

   Luetteloruudun loppuun ilmestyy tyhjä rivi.

3. Kirjoita **Tekstin &väri...**, paina SARKAINNÄPPÄINTÄ ja kirjoita **mnuTextColorItem**.

   Kello-valikossa on nyt komento nimeltä Tekstin väri. Kolme pistettä osoittavat, että komennolla avataan valintaikkuna.

4. Lopeta Menu Editor napsauttamalla OK.

Yleisiä valintaikkunoita ohjaavat tapahtuma-aliohjelmat

Kun haluat avata yleisen valintaikkunan ohjelmassa, sinun pitää kutsua yleistä valintaikkunaobjektia tapahtuma-aliohjelmassa sopivalla objektimetodilla. Sinun on tarvittaessa määritettävä yksi tai useita ominaisuuksia ennen valintaikkunaobjektin kutsumista. Kun käyttäjä on tehnyt valintansa yleisessä valintaikkunassa, tapahtuma-aliohjelma käsittelee ne.

Seuraava tehtäväsi on kirjoittaa ohjelmalauseet mnuOpenItem_Click-aliohjelmaan, joka suoritetaan, kun käyttäjä valitsee Avaa-komennon. Määritä CommonDialog1-objektille Filter-ominaisuus, joka määrää yleisen Open-valintaikkunan tiedostotyypin. (Tarkoitus on luetella vain Windows-metatiedostot.) Avaa sitten Open-valintaikkuna ShowOpen-metodilla. Kun käyttäjä on valinnut tiedoston ja sulkenut valintaikkunan, näytä tiedoston sisältö kuvaobjektissa sijoittamalla Image1-objektin Picture-ominaisuuteen käyttäjän valitseman tiedoston nimi. Ota lopuksi käyttöön Sulje-komento, jotta käyttäjä voi halutessaan poistaa kuvan ohjelman ikkunasta.

Open-aliohjelman muokkaaminen

Näytä lauseet -painike

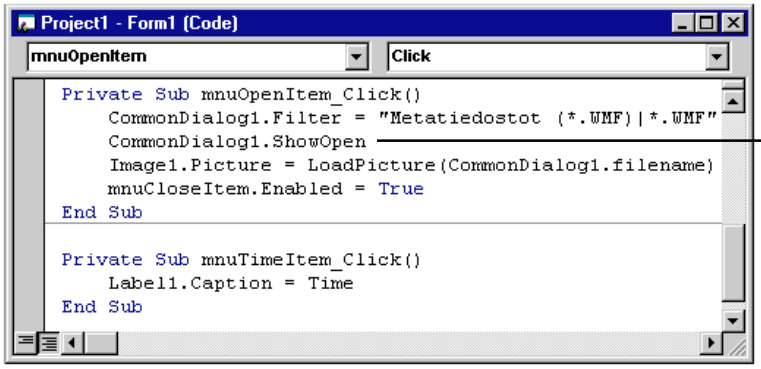
1. Napsauta Project-ikkunan Näytä lauseet -painiketta.
2. Napsauta avattavaa objektiluetteloa ja valitse mnuOpenItem-objekti.

   Tapahtuma-aliohjelma mnuOpenItem_Click ilmestyy Code-ikkunaan.

3. Kirjoita seuraavat ohjelmalauseeet tapahtuma-aliohjelmaan Private Sub- ja End Sub -lauseiden väliin. Sisennä jokaista riviä neljän välilyönnin verran, jotta huomaat niiden kuuluvan tapahtuma-aliohjelmaan. Kirjoita jokainen rivi täsmälleen oikein, ja paina viimeisen rivin jälkeen ALANUOLINÄPPÄINTÄ.

```
CommonDialog1.Filter = "Metatiedostot (*.WMF)|*.WMF"
CommonDialog1.ShowOpen
Image1.Picture = LoadPicture(CommonDialog1.FileName)
mnuCloseItem.Enabled = True
```

Project-ikkunan pitäisi näyttää tällaiselta:



Filter-ominaisuus määrää, millaiset tiedostot näkyvät Open-valintaikkunassa.

Tapahtuma-aliohjelman kolme ensimmäistä lausetta viittaavat CommonDialog1-objektin kolmeen eri ominaisuuteen. Ensimmäinen lause määrää Filter-ominaisuudella, että tiedostoluettelossa näkyy vain Windows-metatiedostot. Tämä on tärkeää Open-valintaikkunassa, koska kuvaobjekti pystyy esittämään kuudenlaisia tiedostoja: bittikarttoja (.bmp), Windows-metatiedostoja (.wmf), kuvakkeita (.ico), osoittimia (.cur), JPEG-tiedostoja (.jpg) ja GIF-tiedostoja (.gif), kuten harjoituksessa 2 kerrottiin. Jos yrität ladata esimerkiksi .txt-tiedoston kuvaobjektiin, aiheutuu ajonaikainen virhe.

Voit lisätä Filter-ominaisuuden luetteloon muita vaihtoehtoja erottelemalla ne pystyviivoin (|). Esimerkiksi lause

```
CommonDialog1.Filter = "Bittikartat (*.BMP)|*.BMP|Metatiedostot (*.WMF)|*.WMF"
```

sallii bittikartta- ja metatiedostojen luettelemisen Open-valintaikkunassa.

Tapahtuma-aliohjelman toinen lause avaa Open-valintaikkunan. Kukin yleinen valintaikkuna avataan eri objektimetodilla; Open-valintaikkunaa vastaava metodi on ShowOpen. Muut yleisiä valintaikkunoita avaavat metodit on lueteltu aiemmin tässä harjoituksessa. Tämä on tapahtuma-aliohjelman tärkein lause. Koska komennon nimi on Open, aliohjelman pitää näyttää Open-valintaikkuna ja käsitellä tulokset.

Kolmas lause käsittelee tiedostoa, jonka käyttäjä valitsi valintaikkunasta. Kun käyttäjä on valinnut levyaseman, kansion ja tiedoston ja napsauttanut OK, metodi palauttaa koko polkunimen ohjelmaan CommonDialog1.FileName-ominaisuuden kautta. Sitten ohjelma kopioi valitun Windows-metatiedoston Image1-objektiin LoadPicture-funktiolla eli aliohjelmalla, joka lukee grafiikkatiedostoja.

Aliohjelman viimeinen lause palauttaa käyttöön Tiedosto-valikon komennon Sulje. Koska tiedosto on avattu, Sulje-komennon pitää olla käytettävissä, jota käyttäjä voi sulkea tiedoston.

Kirjoita seuraavaksi lauseet mnuTextColorItem_Click-aliohjelmaan, joka suoritetaan, kun Kello-valikosta valitaan komento Tekstin väri.

TextColor-aliohjelman muokkaaminen

1. Valitse objektiluettelosta mnuTextColorItem.

   Tekstin väri -komentoa vastaava tapahtuma-aliohjelma ilmestyy Code-ikkunaan.

2. Kirjoita tapahtuma-aliohjelman Private Sub- ja End Sub -lauseiden väliin seuraavat ohjelmalauseet (neljän välilyönnin verran sisennettyinä):

```
CommonDialog1.Flags = &H1&
CommonDialog1.ShowColor
Label1.ForeColor = CommonDialog1.Color
```

Värivalintojen ohjaaminen käyttämällä lippuja

Tapahtuma-aliohjelma mnuTextColorItem_Click käyttää hyväkseen yleisen valintaikkunaobjektin ominaisuuksia ja metodeja. Ensimmäinen lause määrittää Flags-ominaisuudeksi &H1&. Tämä heksadesimaalikoodi merkitsee sitä, että Color-valintaikkunassa näkyy joukko vakiovärejä, muokatut värit ovat käytettävissä ja oletusväri on korostettu. Seuraavassa taulukossa on selitetty Flags-ominaisuuden neljä mahdollista arvoa.

Flags-ominaisuus määrää, millainen Color-valintaikkuna avautuu.

Lippu	Merkitys
&H1&	Näyttää normaalin Color-valintaikkunan (josta voidaan valita myös mukautettuja värejä) ja määrää, että nykyinen väri on oletusväri.
&H2&	Näyttää normaalin ja mukautetun Color-valintaikkunan.
&H4&	Näyttää normaalin Color-valintaikkunan, jossa Define Custom Colors -painike on poistettu käytöstä.
&H8&	Lisää Color-valintaikkunaan Help-painikkeen.

Voit käyttää näiden arvojen mitä tahansa yhdistelmää, kun valmistelet Color-ikkunan ennen avaamista. Yhdistä kaksi tai useampia arvoja Or-operaattorilla. Esimerkiksi lause

```
CommonDialog1.Flags = &H1& Or &H8&
```

näyttää saman Color-valintaikkunan kuin aikaisemmin, mutta lisää siihen Help-painikkeen.

Tapahtuma-aliohjelman toinen lause avaa Color-valintaikkunan kutsumalla ShowColor-metodia ja kolmas sijoittaa valitun värin Label1-objektin ForeColor-ominaisuuteen. Muistanet Label1-objektin aiemmin tässä harjoituksessa käsitellystä MyMenu-ohjelmasta — sillä näytettiin ohjelmassa tämänhetkinen aika ja päivämäärä. Käytä Color-valintaikkunan palauttamaa väriä, kun valitset tekstiobjektin tekstin värin.

HUOMAA Color-valintaikkunan avulla voidaan valita käyttöliittymän minkä tahansa sellaisen osan väri, jonka väri on vaihdettavissa. Muita mahdollisuuksia ovat lomakkeen taustaväri, lomakkeeseen sijoitettujen kuvioiden värit sekä objektien edusta- ja taustavärit.

Seuraavaksi sinun pitää kirjoittaa ohjelmalauseet mnuCloseItem_Click-aliohjelmaan, joka sulkee kuvaobjektissa näkyvän tiedoston käyttäjän valittua Tiedosto-valikosta komennon Sulje.

Close-aliohjelman muokkaaminen

1. Napsauta Code-ikkunan objektiluetteloa ja valitse siitä mnuCloseItem-objekti.

   Tiedosto Sulje -komentoa vastaava tapahtuma-aliohjelma ilmestyy Code-ikkunaan.

2. Kirjoita tapahtuma-aliohjelman Private Sub- ja End Sub -lauseiden väliin seuraavat ohjelmalauseet (neljän välilyönnin verran sisennettyinä):

```
Image1.Picture = LoadPicture("")
mnuCloseItem.Enabled = False
```

LoadPicture-funktion kutsu, jossa lainausmerkkien väli on tyhjä, tyhjentää kuvan tai kuvakehyksen.

   Ensimmäinen lause sulkee avoimen Windows-metatiedoston lataamalla Image1-objektiin tyhjän kuvan. (Tällä menetelmällä tyhjennetään kuvaobjekteja.) Toinen lause muuttaa Tiedosto-valikossa olevan Sulje-komennon himmennetyksi, koska tiedosto ei ole enää avoinna.

Kirjoita vielä mnuExitItem_Click-aliohjelma, joka lopettaa ohjelman suorituksen, kun käyttäjä valitsee Tiedosto-valikosta komennon Lopeta. Tämä on ohjelman viimeinen tapahtuma-aliohjelma.

Exit-aliohjelman muokkaaminen

1. Valitse objektiluettelosta mnuExitItem.

   Tiedosto Lopeta -komentoa vastaava tapahtuma-aliohjelma ilmestyy Code-ikkunaan.

2. Kirjoita tapahtuma-aliohjelman Private Sub- ja End Sub -lauseiden väliin seuraava lause (neljän välilyönnin verran sisennettynä):

   ```
   End
   ```

   Tämä lause päättää ohjelman suorituksen, kun käyttäjä haluaa lopettaa. (Sen pitäisi olla sinulle jo tuttu.)

3. Sulje Code-ikkuna.

Tallenna projekti -painike

4. Tallenna valmis projekti napsauttamalla työkalurivin Tallenna projekti -painiketta.

   Visual Basic tallentaa muutokset, jotka olet tehnyt MyDialog-ohjelman lomake- ja projektitiedostoihin.

Käynnistä MyDialog-ohjelma ja kokeile luomiasi valikoita ja valintaikkunoita.

MyDialog-ohjelman suorittaminen

Käynnistä-painike

Valmis Dialog-ohjelma on kansiossa \Vb5Sbs\Less03.

1. Napsauta työkalurivin Käynnistä-painiketta.

   Ohjelma käynnistyy ja näyttää valikkorivillä valikot Tiedosto ja Kello.

2. Valitse lomakkeen Tiedosto-valikosta komento Avaa.

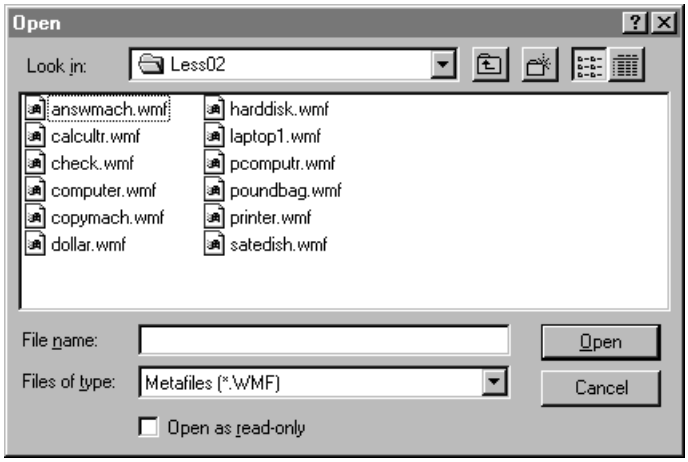
   Yleinen Open-valintaikkuna avautuu. Huomaa, että tiedostotyyppi-luettelossa on teksti Metatiedostot (*.WMF). Määritit sen lauseella

   ```
   CommonDialog1.Filter = "Metatiedostot (*.WMF)|*.WMF"
   ```

   mnuOpenItem_Click-aliohjelmassa. Lainausmerkein ympäröidyn tekstin ensimmäinen osa — Metatiedostot (*.WMF) — on teksti, joka näkyy tiedostotyyppiluettelossa. Toinen osa — *.WMF — määrää valintaikkunassa lueteltavien tiedostojen oletustarkenteen.

3. Avaa kiintolevyn kansio \Vb5Sbs\Less02.

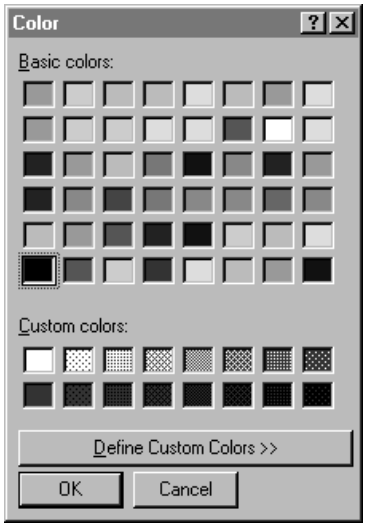
   Kansioon Less02 tallennetut Windows-metatiedostot näkyvät tiedosto-luettelossa seuraavan kuvan mukaisina.



4 Kaksoisnapsauta tiedostoa pcomputr.wmf.
Kuvakehykseen tulee mikrotietokoneen kuva.
Kokeile seuraavaksi Kello-valikon komentoja.

5 Valitse Kello-valikosta komento Aika.
Tämänhetkinen kellonaika ilmestyy tekstikehykseen.

6 Valitse Kello-valikosta komento Tekstin väri.
Yleinen Color-valintaikkuna avautuu tämän näköisenä:



Color-valintaikkunassa on toimintoja, joilla vaihdetaan ohjelman näyttämän kellonajan väri. Nykyinen väri eli musta on valmiiksi valittu.

7 Valitse vaaleansininen väri ja napsauta OK-painiketta.

Color-valintaikkuna sulkeutuu, ja kellonajan numerot muuttuvat vaaleansinisiksi.

8 Valitse Kello-valikosta komento Päivä.

Ohjelma näyttää tämänhetkisen päivämäärän vaaleansinisenä. Kun tekstin väri on vaihdettu, uusi väri pysyy voimassa, kunnes väri vaihdetaan uudelleen tai ohjelma lopetetaan.

9 Napsauta Tiedosto-valikkoa.

Huomaa, että Sulje-komento on nyt käytettävissä. Se aktivoitiin mnuOpenItem_Click-aliohjelmassa lauseella mnuCloseItem.Enabled = True.

10 Sulje Windows-metatiedosto painamalla S.

Tiedosto sulkeutuu, ja mikrotietokoneen kuva katoaa.

11 Napsauta Tiedosto-valikkoa.

Sulje-komento on himmennetty eli ei käytettävissä, koska kuvakehyksessä ei ole kuvaa.

12 Valitse Lopeta-komento.

MyDialog-ohjelman suoritus päättyy, ja Visual Basic -ohjelmointiympäristö tulee näkyviin.

Nyt olet oppinut monia tärkeitä komentoja ja menetelmiä, joilla lisätään ohjelmiin valikoita ja valintaikkunoita. Kunhan opit lisää ohjelmalauseista, pystyt soveltamaan näitä taitoja omiin ohjelmiisi.

Normaalista poikkeavien valintaikkunoiden lisääminen ohjelmiin

Entä jos tarvitset ohjelmassasi valintaikkunan, joka on erilainen kuin viisi edellä kuvattua vakiovalintaikkunaa? Ei hätää — mutta joudut tekemään hieman enemmän suunnittelutyötä. Kuten seuraavissa harjoituksissa opit, Visual Basic -ohjelma voi käyttää tietojen syöttämiseen ja näyttämiseen useitakin lomakkeita. Kun teet mukautetun valintaikkunan, sinun on lisättävä ohjelmaan uusia lomakkeita ja syöttö- ja tulostusobjekteja sekä käsiteltävä valintaikkunan osien napsautukset ohjelmalauseilla. Näitä menetelmiä kuvataan harjoituksessa 7. Seuraavassa harjoituksessa opit käyttämään kahta kätevää valintaikkunaa, jotka on erityisesti suunniteltu tekstin syöttämistä (InputBox) ja tekstin näyttämistä (MsgBox) varten. Nämä valintaikkunat auttavat ylittämään kuilun, joka on yleisten ja itse tehtyjen valintaikkunoiden välissä.

Lisätietoja: Pikanäppäinten lisääminen valikoihin

Menu Editorilla voidaan määrittää valikkokomentojen pikanäppäimiä. Nämä ovat näppäinyhdistelmiä, joita painamalla käyttäjä valitsee komennon valikoita avaamatta. Lähes jokaisessa Windows-sovelluksessa (myös Visual Basicissa) on mahdollista kopioida valittu teksti tai muu tieto Leikepöydälle painamalla CTRL+C. Lisää seuraavaksi pikanäppäimet MyDialog-ohjelman Kello-valikkoon.

Pikanäppäinten lisääminen Kello-valikkoon

1. Napsauta työkalurivin Menu Editor -painiketta.

   Menu Editor käynnistyy.

Menu Editor -painike

2. Valitse valikkoluettelosta komento Aika.

   Aika-komennon otsikko ja nimi ilmestyvät valintaikkunaan. Määritä pikanäppäin valitsemalla avattavasta Shortcut-luetteloruudusta sopiva näppäinyhdistelmä. Valitse Aika-komennon pikanäppäimeksi Ctrl+T.

3. Napsauta Shortcut-luetteloruutua, vieritä luetteloa alaspäin ja valitse Ctrl+T.

   Ctrl+T on nyt määritetty Aika-komennon pikanäppäimeksi, ja näppäinyhdistelmä näkyy valikkoluettelossa.

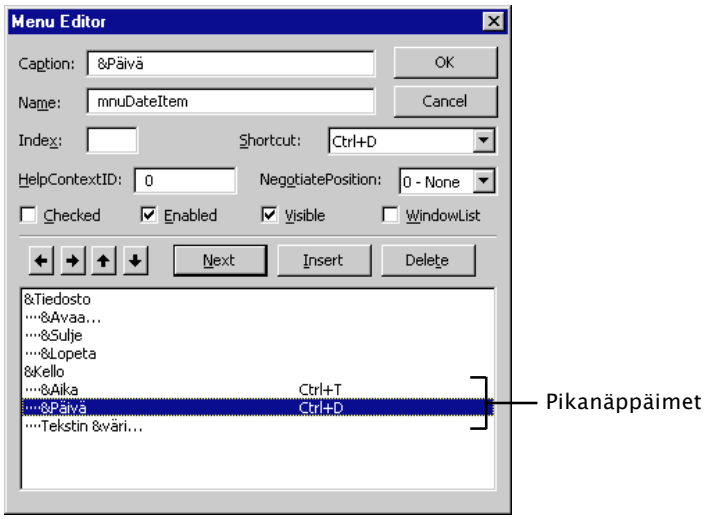
4. Napsauta Next-painiketta.

   Päivä-komennon otsikko ja nimi ilmestyvät valintaikkunaan. Valitse komennon pikanäppäimeksi Ctrl+D.

Et voi määrittää valikon nimelle pikanäppäintä.

5 Napsauta avattavaa Shortcut-luetteloruutua ja valitse Ctrl+D.

Menu Editorin pitäisi näyttää tällaiselta:



6 Lopeta Menu Editor napsauttamalla OK.

Käynnistä ohjelma ja kokeile pikanäppäimiä.

Käynnistä-painike

7 Napsauta työkalurivin Käynnistä-painiketta.

8 Valitse Aika-komento painamalla CTRL+T.

Ohjelma näyttää tämänhetkisen ajan.

9 Valitse Päivä-komento painamalla CTRL+D.

Ohjelma näyttää tämänhetkisen päivämäärän.

10 Napsauta Kello-valikkoa.

Pikanäppäinten nimet näkyvät Aika- ja Päivä-komentojen vieressä. Visual Basic lisää nämä valikkoon, kun määrität pikanäppäimet Menu Editorilla.

11 Valitse Tiedosto-valikosta komento Lopeta.

Ohjelman suoritus päättyy, ja ohjelmointiympäristö tulee näkyviin.

12 Tallenna pikanäppäinten määritykset levylle napsauttamalla Tallenna projekti -painiketta.

Tallenna projekti -painike

Jos haluat harjoitella ohjelmointitaitojasi

Lue asiakirja Review & Practice 1, joka on tallennettu kansion \Vb5Sbs\Review tiedostoon Rev1.doc. Se sisältää vaiheittain eteneviä tehtäviä, joilla voit testata ohjelmointitaitosi ja kerrata lukujen 1–3 sisällön. Tehtävissä muodostetaan bittikarttojen katselemiseen tarkoitettu apuohjelma nimeltä Bit Map Browser.

Jos haluat jatkaa seuraavaan harjoitukseen

➤ Jätä Visual Basic käyntiin ja siirry harjoitukseen 4.

Jos haluat lopettaa Visual Basicin

➤ Valitse File-valikosta komento Exit.

Jos näyttöön tulee Save-valintaikkuna, napsauta Yes.

Harjoituksen tiivistelmä

Tehtävä	Ohje	Painike
Luoda valikko tai komento	Napsauta Menu Editor -painiketta ja määritä valikon tai komennon otsikko, nimi ja sijainti.	
Lisätä valikkoon tai komentoon valintakirjain	Käynnistä Menu Editor, valitse valikko tai komento ja napsauta Caption-teksti-ruutua. Lisää valintakirjaimena käytettävän kirjaimen edelle et-merkki (&).	
Lisätä valikko-komentoon pikanäppäin	Käynnistä Menu Editor ja valitse valikko. Valitse pikanäppäin avattavasta Shortcut-luetteloruudusta.	
Muuttaa valikoiden tai komentojen järjestystä	Käynnistä Menu Editor. Valitse siirrettävä valikko tai komento ja siirrä sitä napsauttamalla ylä- tai alanuolipainiketta.	
Käyttää ohjelmassa vakiovalintaikkunaa	Lisää lomakkeeseen yleinen valintaikkuna-objekti ja tuo valintaikkuna näkyviin kutsumalla ohjelmalauseella yhtä viidestä yleisestä valintaikkunametodista.	
Poistaa valikko-komento käytöstä	Käytä Menu Editoria ja poista komentoon liittyvän Enabled-valintaruudun valinta.	
Ottaa valikko-komento käyttöön ohjelmallisesti	Käytä ohjelmalausetta `mnuCloseItem.Enabled = True` mutta kirjoita kohtaan *mnuCloseItem* komennon nimi.	
Poistaa kuva	Käytä ohjelmalausetta `Image1.Picture = LoadPicture("")`	

Kun tarvitset tietoja aiheesta	Valitse Visual Basicin Help-valikosta Microsoft Visual Basic Help Topics, valitse Index-välilehti ja...
Menu Editor	Kirjoita **Menu Editor**.
CommonDialog-objekti	Kirjoita **CommonDialog control**.
Yleiset valintaikkunat ja niiden metodit	Kirjoita **ShowOpen**, **ShowSave**, **ShowPrinter**, **ShowFont** tai **ShowColor**.

Tietoja seuraavasta harjoituksesta

Osassa 2 (harjoituksissa 4–6) kuvataan ohjelmoinnin perusteita, jotta osaat tehdä tehokkaita ohjelmia. Harjoituksessa 4, näet, miten ohjelmissa luodaan muuttujia. Samalla opit käyttämään operaattoreita ja matemaattisia funktioita.

Osa

2

Ohjelmoinnin perusteita

Harjoitus 4
Visual Basicin muuttujat ja operaattorit 107

Harjoitus 5
Ehtorakenteet 135

Harjoitus 6
Silmukat ja ajastimet 161

Harjoitus

4

Visual Basicin muuttujat ja operaattorit

Arvioitu aika
50 min.

Tässä harjoituksessa opit

- tallentamaan ohjelman tietoja muuttujiin
- lukemaan syöttötietoja InputBox-funktiolla
- näyttämään ilmoituksia MsgBox-funktiolla
- käyttämään matemaattisia operaattoreita ja funktioita kaavoissa.

Osassa 1 opit luomaan Visual Basic -ohjelman käyttöliittymän sekä kirjoittamaan ja käynnistämään ohjelman Visual Basic -ohjelmointiympäristössä. Kolmessa seuraavassa harjoituksessa opit lisää Visual Basic -ohjelmista — lauseista ja avainsanoista, joista ohjelma muodostuu. Osan 2 jälkeen olet valmis siirtymään vaativampiin aiheisiin.

Tässä harjoituksessa näytän, miten ohjelman tietoja tallennetaan tilapäisesti muuttujiin ja miten matemaattisilla operaattoreilla suoritetaan laskutoimituksia, kuten yhteen- ja kertolaskuja. Kerron myös, miten matemaattisilla funktioilla käsitellään lukuja ja miten InputBox- ja MsgBox-funktiot kysyvät ja näyttävät tietoja valintaikkunoissa.

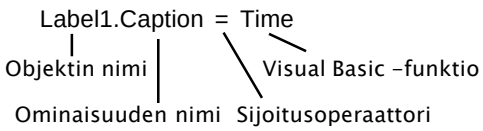
Visual Basic -ohjelmalauseen rakenne

Ohjelmalause on toimintaohje Visual Basic -kääntäjälle.

Opit harjoituksessa 1, että Visual Basic -ohjelman riviä sanotaan ohjelmalauseeksi. *Ohjelmalause* on mikä tahansa Visual Basicin avainsanojen, ominaisuuksien, funktioiden, operaattoreiden ja symbolien yhdistelmä, joista muodostuu Visual Basic -kääntäjän hyväksymä toimintaohje. Täydellinen ohjelmalause saattaa sisältää vain yhden avainsanan. Esimerkiksi

```
Beep
```

on lause, joka antaa äänimerkin tietokoneen kaiuttimella. Toisaalta lause voi olla monen osan yhdistelmä. Esimerkiksi seuraava lause sijoittaa tietokoneen kellosta luetun ajan tekstiobjektin Caption-ominaisuuteen:



Sääntöjä, joita on noudatettava ohjelmalausetta rakennettaessa, sanotaan *kieliopiksi* tai *syntaksiksi*. Monet Visual Basicin kielioppisäännöistä ovat samoja kuin Basic-ohjelmointikielen aiemmissa versioissa tai muissa ohjelmointikielissä. Jotta osaisit kirjoittaa säännönmukaisia ohjelmalauseita, sinun on tunnettava ohjelmointikielen tärkeimpien osien kielioppi ja käytettävä näitä osia oikealla tavalla tietojen käsittelemiseen. Onneksi Visual Basic tekee raskaimmat työt puolestasi, joten ohjelmalauseiden kirjoittamiseen tarvittava aika jää suhteellisen lyhyeksi, ja voit käyttää tulokset uudelleen muissa ohjelmissa.

Seuraavissa harjoituksissa opit Visual Basicin tärkeimmät avainsanat ja ohjelmalauseet. Voit huomata, että ne täydentävät sopivasti jo oppimiasi ohjelmointitaitoja ja auttavat ohjelmien kirjoittamisessa. Muuttujat ja tietotyypit, joita kuvaan ensin, ovat jokaisen ohjelman olennaisia osia.

Tietojen tallentaminen muuttujiin

Muuttuja on tilapäinen säilytyspaikka ohjelman tiedoille. Voit käyttää ohjelmassa yhtä tai useita muuttujia, ja ne voivat sisältää tekstiä, lukuja, päivämääriä tai ominaisuuksia. Muuttujat ovat käteviä, koska voit antaa lyhyen ja helposti muistettavan nimen tiedolle, jota aiot käsitellä. Muuttuja voi sisältää käyttäjän antaman tiedon, laskutoimituksen tuloksen tai lomakkeessa esitettävän tekstin. Muuttujat ovat lyhyesti sanoen välineitä, joihin tallennetaan melkein millaisia tietoja tahansa.

Voit käyttää muuttujaa suoraan mainitsemalla sen nimen ohjelmalauseessa tai *esitellä* sen ensin. Kuvaan muuttujien varaamista eli esittelemistä kahdessa seuraavassa kohdassa.

Muuttujien varaaminen: Dim-lause

Dim-lause varaa tilaa muuttujalle.

Jos haluat esitellä muuttujan erikseen, kirjoita sen nimi Dim-lauseeseen (Dim on lyhenne sanasta *dimension*) ennen kuin käytät muuttujaa ensimmäisen kerran, tavallisesti tapahtuma-aliohjelman alkuun. Dim-lause varaa muuttujalle muistitilaa ohjelman suorituksen aikana ja ilmaisee Visual Basicille, millaisia tietoja muuttujaan on tarkoitus tallentaa. Esimerkiksi seuraava lause varaa tilaa muuttujalle nimeltä Sukunimi:

```
Dim Sukunimi
```

Voit määrittää muuttujan nimen jälkeen tyypin, jos haluat (kuvaan perustietotyyppejä myöhemmin tässä luvussa). Valitsemalla muuttujan tyypin etukäteen voit vaikuttaa siihen, paljonko tilaa ohjelma varaa muuttujalle. Jos muuttujaan on tarkoitus sijoittaa vain pieniä lukuja, joissa ei ole desimaaliosaa, eli kokonaislukuja, esittele muuttuja kokonaisluvuksi, niin säästät hieman muistitilaa. Normaalisti Visual Basic varaa tilaa *muunnelmatyyppiselle* muuttujalle, jonka tietojen kokoa ja muotoa ei ole rajoitettu. Yleiskäyttöinen muunnelmamuuttuja on erittäin joustava ja ehkä ainoa muuttujatyyppi, jota tarvitset ohjelmissasi.

Kun olet esitellyt muuttujan, voit sijoittaa siihen tietoja ohjelmassa. Esimerkiksi seuraava ohjelmalause sijoittaa Sukunimi-muuttujaan sukunimen ”Jaakkola :

```
Sukunimi = "Jaakkola"
```

Muuttujaan tallennetaan tietoja sijoitusoperaattorilla (=).

Tämän sijoituksen jälkeen ohjelmassa voidaan käyttää Sukunimi-muuttujaa sukunimen ”Jaakkola” sijasta. Esimerkiksi sijoituslause

```
Label1.Caption = Sukunimi
```

näyttää lomakkeen ensimmäisen tekstiobjektin (Label1) kohdalla tekstin *Jaakkola*.

Muuttujien esitteleminen ilman Dim-lausetta

Voit esitellä muuttujan myös ilman Dim-lausetta; tätä sanotaan *oletetuksi esittelyksi*. Sinun tarvitsee vain käyttää muuttujaa sellaisenaan ja jättää Dim-lause kokonaan pois:

```
Sukunimi = "Kaarle  V"
```

Oletettu esittely säästää hieman aikaa, koska sinun ei tarvitse kirjoittaa Dim-lausetta. Toisaalta se ei ole suositeltava, koska sinun ei tarvitse järjestellä eikä luetella muuttujia etukäteen ja koska Visual Basic ei pysty antamaan virheilmoitusta, jos kirjoitat muuttujan nimen väärin (katso seuraavaa huomautusta). Tässä kirjassa esittelen muuttujia molemmilla tavoilla.

HUOMAA Jos päätät esitellä muuttujat aina Dim-lauseella, sinun kannattaa lisätä Option Explicit -lause käynnistyslomakkeen esittely-osaan. Voit antaa Visual Basicin lisätä sen automaattisesti jokaiseen uuteen projektiin: Valitse Tools-valikosta komento Options, napsauta Editor-välilehteä ja valitse Require Variable Declaration -valintaruutu.

Kun ohjelmassa on Option Explicit -lause, Visual Basic ilmoittaa virheestä aina löytäessään ohjelmasta muuttujan, jota ei ole erikseen esitelty. Virheen syy on todennäköisesti se, että muuttujan nimi on kirjoitettu väärin. Jos pelkäät kirjoitusvirheitä, tämä lause helpottaa niiden löytämistä.

Muuttujien käyttäminen ohjelmassa

Muuttujan arvo voi säilyä samana ohjelman suorituksen ajan tai vaihtua useita kertoja, aivan miten haluat. Seuraavassa näet, miten muuttujaan sijoitetaan välillä teksti ja välillä luku ja miten muuttuja sijoitetaan objektin ominaisuudeksi.

Muuttujan arvon muuttaminen

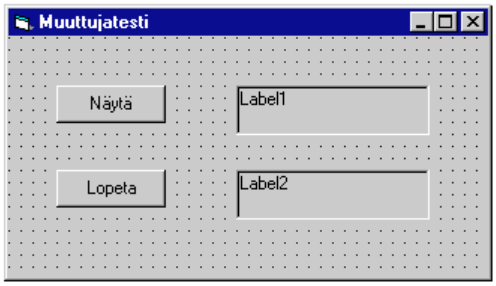
1. Käynnistä Visual Basic.
2. Valitse File-valikosta komento Open Project.

   Open Project -valintaikkuna avautuu.
3. Avaa kansiosta \Vb5Sbs\Less04 malliprojekti VarTest.

   Muuttujatesti-projekti avautuu ohjelmointiympäristöön. Muuttujatesti on tynkäohjelma — se sisältää lomakkeen, jossa on tekstiruudut ja painikkeet tietojen näyttämistä varten, mutta vain vähän ohjelmalauseita. Tehtäväsi on lisätä ohjelmalauseet.
4. Valitse lomakkeen nimi Project-ikkunasta ja napsauta Project-ikkunan Näytä objekti -painiketta.

   Muuttujatesti-lomake tulee näkyviin seuraavan kuvan mukaisena.

Näytä objekti -painike



Lomakkeessa on kaksi tekstiruutua ja kaksi komentopainiketta. Voit tuoda tekstiruutuihin tietoja käyttämällä muuttujia.

5 Kaksoisnapsauta Näytä-painiketta.

Tapahtuma-aliohjelma Command1_Click ilmestyy Code-ikkunaan.

6 Kirjoita seuraavat ohjelmalauseet, jotka esittelevät Sukunimi-muuttujan ja käyttävät sitä:

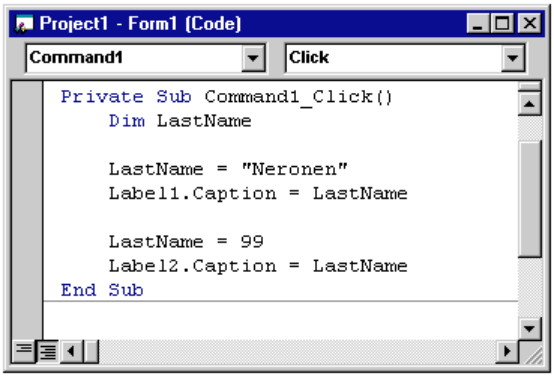
```
Dim LastName

LastName = "Neronen"
Label1.Caption = LastName

LastName = 99
Label2.Caption = LastName
```

Muuttujilla voidaan siirtää tietoja ominaisuuksiin.

Ohjelmalauseet jakautuvat kolmeen ryhmään. Ensimmäinen lause esittelee LastName-muuttujan Dim-lauseella. Koska tyyppiä ei ole määritetty, muuttujasta tulee muunnelmatyyppinen eli siihen voidaan tallentaa tekstiä tai lukuja. Toinen ja kolmas lause sijoittavat LastName-muuttujaan nimen ”Neronen” ja näyttävät sen lomakkeen ensimmäisessä tekstiruudussa. Tämä havainnollistaa yhtä muuttujien tavallisimmista tehtävistä eli tietojen sijoittamista ominaisuuteen.

Neljäs lause sijoittaa LastName-muuttujaan luvun 99. Teksti katoaa muuttujasta ja korvautuu luvulla. Luku ei ole lainausmerkeissä. Tekstit eli merkkijonot on ympäröitävä lainausmerkein, mutta lukuja ei. Jos luvun ympärillä on lainausmerkit, se muuttuu merkkijonoksi, ja sitä ei voi käyttää matemaattisissa laskutoimituksissa.

Project-ikkunan pitäisi näyttää tällaiselta:



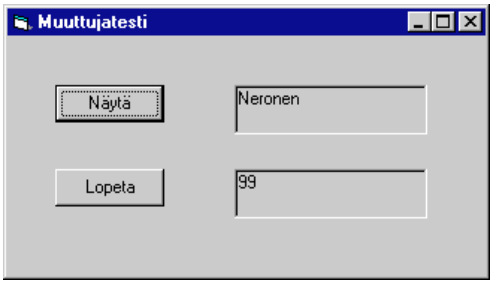
Käynnistä-painike

7 Käynnistä ohjelma napsauttamalla työkalurivin Käynnistä-painiketta.

Ohjelma käynnistyy ohjelmointiympäristön alaisuudessa.

8 Napsauta Näytä-painiketta.

Ohjelma esittelee muuttujan, sijoittaa siihen kahdenlaisia tietoja ja kopioi tiedot vuorotellen lomakkeen kahteen tekstiruutuun. Ohjelman tulokset näkyvät seuraavassa kuvassa.



9 Lopeta ohjelma napsauttamalla Lopeta-painiketta.

Ohjelma pysähtyy, ja ohjelmointiympäristö palaa näkyviin.

10 Tallenna muokkaamasi lomake nimellä **MyVarTest.frm** valitsemalla File-valikosta komento Save VarTest.frm As. Tallenna myös projekti nimellä **MyVarTest.vbp** valitsemalla File-valikosta komento Save Project As.

Muuttujien nimeämissääntöjä

Muuttujien nimeäminen voi tuottaa hieman vaikeuksia, koska nimien pitäisi olla lyhyitä, mutta kuvaavia ja helppoja muistaa. Noudata seuraavia nimeämissääntöjä sekaannusten välttämiseksi:

- Aloita muuttujan nimi kirjaimella; tämä on Visual Basicin asettama vaatimus. Muuttujan nimessä saa olla enintään 255 merkkiä, ja pisteet ovat kiellettyjä.
- Valitse kuvaava nimi, jossa on tarvittaessa useita sanoja. Esimerkiksi nimi MyyntiveroProsentti on paljon selkeämpi kuin Vero tai Prosentti.
- Käytä isoja ja pieniä kirjaimia ja numeroita, jos haluat. Muuttujan nimen jokainen sana on tapana aloittaa isolla kirjaimella, esimerkiksi PuhelinKotiin.
- Älä käytä Visual Basicin avainsanoja, objektien nimiä tai ominaisuuksien nimiä muuttujien niminä.

Syöttötietojen tallentaminen muuttujaan

Voit lukea käyttäjän antamia tietoja kätevästi käyttäen InputBox-funktiota ja muuttujaa.

Yksi muuttujan tärkeimpiä tehtäviä on tallentaa käyttäjän antamat tiedot. Voit usein käyttää objektia (esimerkiksi tiedostoluetteloa tai tekstiruutua) tietojen lukemiseen, mutta haluat ehkä viestiä suoraan käyttäjän kanssa ja tallentaa syöttötiedot muuttujaan eikä ominaisuuteen. Valitse tällöin InputBox-funktio, joka avaa näyttöön valintaikkunan ja tallentaa käyttäjän kirjoittaman tekstin muuttujaan. Kokeile tätä seuraavassa esimerkissä.

Tietojen lukeminen InputBox-funktiolla

1. Valitse File-valikosta komento Open Project.

   Open Project -valintaikkuna avautuu.

2. Avaa projekti InputBox, joka on kansiossa \Vb5Sbs\Less04.

   InputBox-projekti avautuu ohjelmointiympäristöön. Input Box on tynkäohjelma — siinä on lomake, joka sisältää painikkeita ja tekstikehyksen tulosten näyttämistä varten, mutta vain vähän ohjelmalauseita.

3. Valitse lomake Project-ikkunasta ja napsauta Project-ikkunan Näytä objekti -painiketta.

   Syöttöruudun testaus -lomake tulee näkyviin. Siinä on yksi tekstikehys ja kaksi komentopainiketta. Lue käyttäjän antamat tiedot InputBox-funktiolla ja näytä ne lomakkeen tekstikehyksessä.

4. Kaksoisnapsauta Syöttöruutu-komentopainiketta.

   Tapahtuma-aliohjelma Command1_Click ilmestyy Code-ikkunaan.

5 Kirjoita seuraavat ohjelmalauseet, jotka esittelevät kaksi muuttujaa ja kutsuvat InputBox-funktiota:

```
Dim Prompt,  FullName
Prompt = "Kirjoita  nimesi."

FullName = InputBox$(Prompt)
Label1.Caption = FullName
```

Tällä kertaa esittelet Dim-lauseella kaksi muuttujaa: Prompt ja FullName. Tapahtuma-aliohjelman toinen lause sijoittaa joukon merkkejä eli *merkkijonon* muuttujaan Prompt. Tätä tekstiä käytetään InputBox-funktion tekstiargumenttina. *Argumentti* on arvo tai lauseke, joka välitetään aliohjelmalle tai funktiolle. Seuraava lause kutsuu InputBox-funktiota ja sijoittaa kutsun tuloksen eli käyttäjän kirjoittaman tekstin muuttujaan FullName. InputBox on Visual Basicin erityisfunktio, joka näyttää valintaikkunan ja pyytää käyttäjältä tietoja. InputBox-funktiolle voidaan antaa kehotemerkkijonon lisäksi muitakin argumentteja. Katso lisätietoja Visual Basicin ohjeista.

Kun InputBox-funktio on palauttanut merkkijonon ohjelmaan, aliohjelman neljäs lause sijoittaa sen Label1-objektin Caption-ominaisuuteen. Tämän jälkeen merkkijono näkyy lomakkeessa.

6 Käynnistä ohjelma napsauttamalla työkalurivin Käynnistä-painiketta.

Ohjelma käynnistyy ohjelmointiympäristön alaisuudessa.

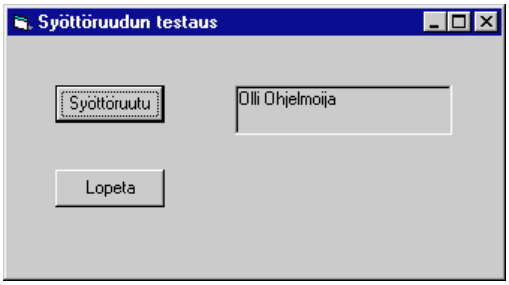
7 Napsauta Syöttöruutu-painiketta.

Käynnistä-painike

Visual Basic suorittaa Command1_Click-aliohjelman, ja InputBox-valintaikkuna avautuu:

8 Kirjoita nimesi ja napsauta OK.

InputBox-funktio palauttaa kirjoittamasi nimen ohjelmaan ja sijoittaa sen FullName-muuttujaan. Ohjelma käyttää muuttujaa näyttäessään nimesi lomakkeessa.



Voit käyttää InputBox-funktiota ohjelmassasi aina halutessasi kysyä käyttäjältä tietoja. Se voi toimia muiden syöttöobjektien kanssa, jos tietojen kulkua ohjelmaan ja ohjelmasta pitää ohjata. Seuraavassa tehtävässä näet, miten samankaltaisella funktiolla näytetään tekstiä valintaikkunassa.

9 Lopeta ohjelma napsauttamalla Lopeta-painiketta.

Ohjelman suoritus päättyy, ja ohjelmointiympäristö palaa näkyviin.

10 Tallenna lomakkeeseen ja projektiin tekemäsi muutokset nimellä **MyInputBox**.

Mikä on funktio?

InputBox on Visual Basicin erityinen avainsana, jota sanotaan funktioksi. *Funktio* on lause, joka suorittaa tehtävän (esimerkiksi kysyy käyttäjältä tietoja tai laskee lausekkeen arvon) ja palauttaa tuloksen ohjelmaan. Funktion palauttama arvo voidaan sijoittaa muuttujaan MyInputBox-ohjelmassa esitetyllä tavalla tai sijoittaa ominaisuuteen, toiseen lauseeseen tai funktioon. Visual Basicin funktioiden toimintaa ohjataan usein käyttämällä yhtä tai useaa argumenttia. Esimerkiksi edellä esitetylle InputBox-funktiolle välitetään Prompt-muuttuja, joka sisältää käyttäjälle näytettävän ohjeen. Kun funktiolla on useita argumentteja, ne erotellaan pilkuin, ja argumenttiryhmä ympäröidään sulkein. Seuraavassa lauseessa on kaksiargumenttinen funktion kutsu. Ensimmäisenä on muuttujan nimi, sitten yhtäläisyysmerkki, kutsuttavan funktion nimi ja sulkein ympäröidyt ja pilkulla erotetut argumentit.

```
FullName = InputBox$(Kehote, Otsikko)
```

Tietojen näyttäminen muuttujalla

Voit näyttää muuttujan sisällön sijoittamalla muuttujan ominaisuuteen, esimerkiksi tekstiobjektin Caption-ominaisuuteen, tai käyttämällä muuttujaa valintaikkunafunktion argumenttina. Esitä tulokset esimerkiksi valintaikkunafunktiolla MsgBox. InputBox-funktion tapaan sekin lukee yhden tai useita argumentteja, ja sen tulos voidaan sijoittaa muuttujaan. MsgBox-funktion kutsu on tällainen

MsgBox-funktio näyttää merkkijonon sisällön valintaikkunassa. Siinä on useita valinnaisia argumentteja.

ValittuPainike = MsgBox(*Ilmoitus*, *PainikkeidenMäärä*, *Otsikko*)

Ilmoitus on valintaikkunassa näytettävä teksti, *PainikkeidenMäärä* on painiketyylin numero (1–5) ja *Otsikko* on ilmoitusikkunan otsikkorivillä näytettävä teksti. *ValittuPainike* on muuttuja, johon sijoitetaan funktion palauttama tulos. Tämä ilmaisee, mitä valintaikkunan painiketta napsautettiin.

Jos vain näytät ilmoituksen, sijoitusoperaattori, *ValittuPainike*-muuttuja ja *PainikkeidenMäärä*-argumentti ovat valinnaisia. Et tarvitse niitä seuraavassa esimerkissä; jos haluat lisätietoja niistä (sekä valintaikkunan painikkeista), etsi Visual Basicin ohjeista hakusanaa *MsgBox*.

Lisää nyt MyInputBox-ohjelmaan MsgBox-funktio, joka näyttää nimen, jonka käyttäjä kirjoitti InputBox-valintaikkunaan.

Ilmoituksen näyttäminen MsgBox-funktiolla

1. Kaksoisnapsauta MyInputBox-lomakkeen Syöttöruutu-painiketta.

   Tapahtuma-aliohjelma Command1_Click ilmestyy Code-ikkunaan. Siinä ovat edellisessä tehtävässä kirjoittamasi lauseet.

2. Valitse tapahtuma-aliohjelmasta seuraava lause (viimeinen rivi):

   ```
   Label1.Caption = FullName
   ```

   Tämä on lause, joka näyttää FullName-muuttujan sisällön tekstikehyksessä.

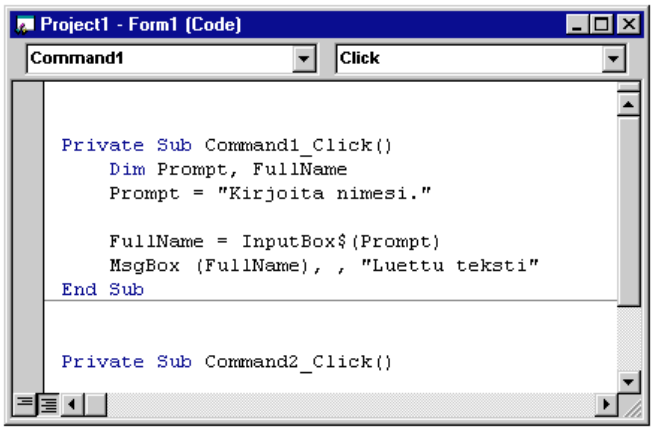
3. Poista lause painamalla DEL.

   Lause katoaa Code-ikkunasta.

4. Kirjoita poistamasi lauseen tilalle tällainen lause:

   ```
   MsgBox (FullName), , "Luettu  teksti"
   ```

   Tämä uusi lause kutsuu MsgBox-funktiota, näyttää FullName-muuttujan sisällön valintaikkunassa ja lisää valintaikkunan otsikkoriville sanat *Luettu teksti*. (Valinnainen argumentti *PainikkeidenMäärä* ja valinnainen muuttuja *ValittuPainike* on jätetty pois.) Tapahtuma-aliohjelman pitäisi näyttää seuraavan kuvan mukaiselta.



HUOMAA Kun *ValittuPainike*-muuttuja puuttuu, sulkeet ympäröivät vain ensimmäisen argumentin.

Käynnistä-painike

5 Napsauta työkalurivin Käynnistä-painiketta.

6 Napsauta Syöttöruutu-painiketta, kirjoita nimesi tekstiruutuun ja napsauta OK.

Ohjelma tallentaa kirjoittamasi merkit FullName-muuttujaan ja näyttää ne sitten ilmoitusikkunassa. Ohjelman pitäisi avata tämänkaltainen ikkuna:

7 Sulje ilmoitusikkuna napsauttamalla OK. Lopeta ohjelma napsauttamalla Lopeta.

Ohjelman suoritus päättyy, ja ohjelmointiympäristö palaa näkyviin.

8 Tallenna lomake ja projekti nimellä **MyMsgBox**, jotta sinulle jää kopio ohjelmasta.

Tietotyyppien käyttö

Jos muuttuja sisältää aina tietynlaisen tiedon, voit parantaa ohjelman suorituskykyä esittelemällä muuttujan oikeantyyppiseksi.

Muunnelma- eli Variant-tyyppi on useimmiten ainoa tarvitsemasi tietotyyppi. Muunnelmamuuttujiin voidaan tallentaa mitä tahansa Visual Basicin (valmiiksi määritetyistä) perustietotyypeistä, ja sen muoto muuttuu automaattisesti. Muunnelmat ovat helppoja käyttää, ja sinun ei tarvitse juuri kiinnittää huomiota muuttujan todelliseen kokoon, kun esittelet sitä. Jos sen sijaan haluat tehdä hyvin nopean ja tiiviin ohjelman, sinun kannattaa määrittää tietotyyppi tilanteen mukaan.

Jos muuttuja sisältää aina esimerkiksi pieniä kokonaislukuarvoja ilman desimaaliosaa, voit säästää muistitilaa ohjelman suorituksen aikana esittelemällä muuttujan kokonaisluvuksi eikä muunnelmaksi. Kokonaislukumuuttuja myös nopeuttaa aritmeettisia operaatioita, joten ohjelman suorituskyky paranee hieman.

Visual Basicin perustietotyypit on lueteltu alla. Seuraavassa tehtävässä pääset kokeilemaan erilaisia tietotyyppejä käytännössä.

Muuttujan tallennuskoko mitataan tavuina; yksi tavu vastaa 8 bittiä (yleensä 1 merkki).

Tietotyyppi	Koko	Arvoalue	Esimerkki
Kokonaisluku (Integer)	2 tavua	−32 768 – 32 767	Dim Linnut% Linnut% = 37
Pitkä kokonaisluku (Long)	4 tavua	−2 147 483 648 – 2 147 483 647	Dim Laina& Laina& = 350000
Yksinkert. tarkkuuden liukuluku (Single)	4 tavua	−3.402823E38 – 3.402823E38	Dim Hinta! Hinta! = 899.99
Kaksoistarkkuuden liukuluku 3.1415926535 (Double)	8 tavua	−1.79769313486232D308 – 1.79769313486232D308	Dim Pii# Pii# =
Valuutta (Currency)	8 tavua	−922337203685477.5808 – 922337203685477.5807	Dim Velka@ Velka@ = 760030.50
Merkkijono (String)	1 tavu/ merkki	0 – 65 535 merkkiä	Dim Koira$ Koira$ = "setteri"
Totuusarvo (Boolean)	2 tavua	True tai False	Dim Tot as Boolean Tot = True
Päivämäärä (Date)	8 tavua	1. tammikuuta 100 – 31. joulukuuta 9999	Dim Syntp as Date Syntp = #3-1-63#
Muunnelma (Variant)	16 tavua luvut); 22 tavua + 1 tavu/merkki (merkkijonot)	Kaikkien muiden tietotyyppien arvoalueet	Dim Summa Summa = 289.13

HUOMAA Voit valita eräitä perustietotyyppejä lisäämällä muuttujan nimen jälkeen tyyppimääritysmerkin. Esimerkiksi kokonaisluku-muuttuja esitellään lisäämällä sen nimen jälkeen prosenttimerkki (%). Siksi kaksi seuraavaa esittelyä merkitsevät Visual Basicissa samaa:

```
Dim I As Integer
Dim I%
```

Perustietotyyppien käyttö ohjelmassa

1 Valitse File-valikosta komento Open Project.

Open Project -valintaikkuna avautuu.

2 Avaa Data-projekti kansiosta \Vb5Sbs\Less04.

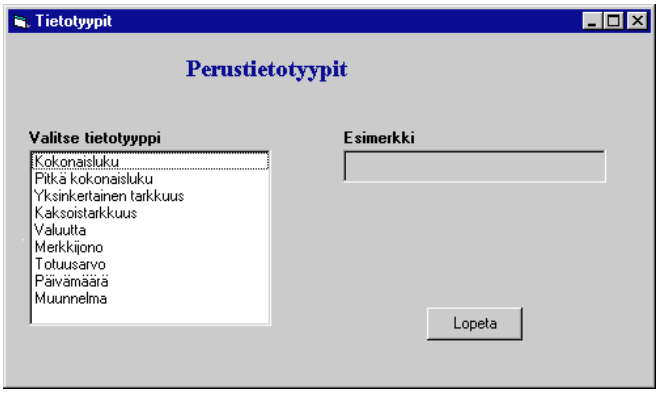
Data-ohjelma esittelee perus-tietotyyppien käyttöä ohjel-massa.

Data-projekti avautuu ohjelmointiympäristöön. Se on valmis Visual Basic -ohjelma, joka esittelee perustietotyyppien toimintaa. Käynnistä ohjelma, jotta näet, millaisilta tietotyypit näyttävät, ja tutki sitten, miten muuttujia esitellään ja käytetään ohjelmassa.

3 Napsauta työkalurivin Käynnistä-painiketta.

Käynnistä-painike

Seuraava sovellusikkuna avautuu:



Data-ohjelmalla voit kokeilla yhdeksää tietotyyppiä, joita ovat muun muassa kokonaisluku, yksinkertaisen tarkkuuden liukuluku ja päivämäärä. Ohjelma näyttää esimerkin kustakin tyypistä, kun valitset sen nimen luetteloruudusta.

4 Valitse luetteloruudusta Kokonaisluku-tyyppi.

Esimerkki-ruutuun ilmestyy numero 37.

5 Valitse luetteloruudusta Päivämäärä-tyyppi.

Esimerkki-ruutuun ilmestyy päivämäärä.

6 Valitse kukin tietotyyppi vuorollaan luettelosta, jotta esimerkki ilmestyy Esimerkki-ruutuun.

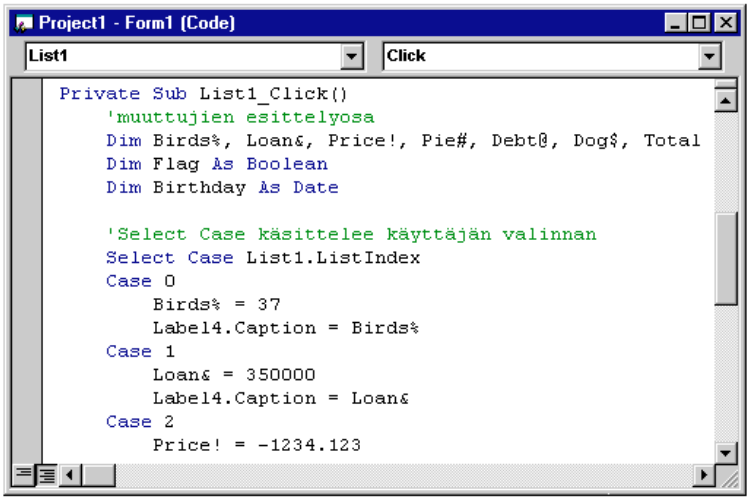
7 Lopeta ohjelma napsauttamalla Lopeta-painiketta.

Tutki, miten perustietotyyppeihin kuuluvia muuttujia esitellään ja käytetään tapahtuma-aliohjelmassa List1_Click.

8 Valitse lomake Project-ikkunasta ja napsauta Näytä objekti -painiketta.

9 Kaksoisnapsauta lomakkeen luetteloruutuobjektia ja suurenna Code-ikkunaa, jotta näet suuremman osan ohjelmasta.

Tapahtuma-aliohjelma List1_Click tulee näkyviin:



Aliohjelman ensimmäiset lauseet esittelevät muuttujia ja määrittävät niiden tietotyypit. Nämä muuttujat ovat aliohjelmassa paikallisia: niillä ei ole merkitystä ohjelman muissa aliohjelmissa. Useat muuttujista on esitelty käyttäen tyyppimääritysmerkkejä, kuten %, # ja @. Näillä symboleilla määritetään muuttujan tietotyyppi Visual Basic -kääntäjää tai ohjelmaasi lukevia käyttäjiä varten.

Aliohjelman seuraavaa osaa sanotaan Select Case -rakenteeksi. Seuraavassa harjoituksessa selitän, miten tämä ohjelmalauseryhmä valitsee yhden useasta vaihtoehdosta. Tällä kertaa sinun pitää vain kiinnittää huomiota siihen, miten Select Case -rakenteen kukin osa sijoittaa esimerkkiarvon yhteen perustietotyyppiseen muuttujaan ja sitten sijoittaa muuttujan Label4-objektin tekstiin. Voit käyttää näitä molempia menetelmiä, kun käsittelet perustietotyyppejä ohjelmissasi.

Date-tietotyyppi on hyödyllinen varsinkin, jos käsittelet usein päivämääriä ja aikoja. Päivämäärä sijoitetaan ristikkomerkein (#) ympäröitynä muuttujaan Birthday ja muotoillaan Format-funktiolla.

HUOMAA Muuttujat voivat olla myös julkisia eli käytettävissä ohjelman kaikissa aliohjelmissa ja moduuleissa. (Moduulit ovat erikoistiedostoja, jotka sisältävät esittelyjä ja aliohjelmia, jotka eivät liity mihinkään tiettyyn lomakkeeseen.) Jotta muuttujan viittausalue olisi julkinen, se on esiteltävä vakiomoduulissa. Tietoja julkisten muuttujien luomisesta vakiomoduuleissa on harjoituksessa 9.

10 Vieritä Code-ikkunaa alaspäin ja tarkastele muuttujien sijoituslauseita huolellisesti.

Voit halutessasi muokata muutamia sijoituslauseita, käynnistää ohjelman uudelleen ja katsoa, miten muutokset vaikuttavat.

11 Kun olet lopettanut, sulje Code-ikkuna.

Jos olet tehnyt muutoksia, jotka haluat säilyttää, napsauta työkalurivin Tallenna projekti -painiketta.

Käyttäjän määrittämät tietotyypit

Visual Basicissa voidaan tehdä myös omia tietotyyppejä. Tästä mahdollisuudesta on hyötyä varsinkin silloin, kun käsitellään tietoja, jotka kuuluvat yhteen mutta ovat eri tyyppiä. Määritä uusi tyyppi Type-lauseella ja esittele siihen kuuluvat muuttujat Dim-lauseella. (Type-lauseen on oltava vakiomoduulin Declarations-osassa; jos tarvitset tietoja vakiomoduuleista, etsi Visual Basicin ohjeista hakusanaa *module*.) Esimerkiksi seuraava lause muodostaa käyttäjän määrittämän tyypin Tyontekija, johon tallennetaan yrityksen työntekijän nimi, syntymäaika ja työhönottopäivä:

```
Type Tyontekija
    Nimi As String
    SyntymaAika As Date
    TyohonottoPaiva As Date
End Type
```

Kun olet määrittänyt uuden tietotyypin, voit käyttää sitä ohjelmassa. Seuraavat lauseet viittaavat uuteen Tyontekija-tyyppiin. Ensimmäinen lause luo Tyontekija-tyyppisen muuttujan nimeltä Tuotepaallikko, ja toinen lause sijoittaa muuttujan Nimi-osaan nimen ”Erkki Kotilainen”:

```
Dim ProductManager As Employee
ProductManager.Name = "Erkki  Kotilainen"
```

Tämähän vaikuttaa aivan ominaisuuden määrittämiseltä, vai kuinka? Visual Basicissa esitetään objektin ja ominaisuuden välinen suhde samalla tavoin kuin käyttäjän määrittämän tyypin ja sen osan välinen suhde.

Vakiot: muuttumatomat muuttujat

Jos ohjelmassa on muuttuja, jonka arvo ei koskaan muutu (esimerkiksi matemaattinen vakio π), sinun kannattaa harkita arvon tallentamista vakioon muuttujan sijasta. *Vakio* on kuvaava nimi luvulle tai merkkijonolle, joka ei muutu. Vakiot ovat hyödyllisiä, koska ne parantavat ohjelman luettavuutta, ehkä säästävät muistia ja helpottavat yleisten muutosten tekemistä. Vakiot toimivat samalla tavoin kuin muuttujat, mutta niiden arvoja ei voi muuttaa ohjelman suorituksen aikana. Ne esitellään Const-avainsanalla näin:

```
Const Pii = 3.14159265
```

Tämä lause luo Pii-nimien vakion, jota voidaan käyttää ohjelmassa piin likiarvon sijasta. Jos haluat vakion olevan käytettävissä ohjelman kaikissa aliohjelmissa, määritä se vakiomoduulissa ja lisää sen edelle avainsana Public. Esimerkki:

```
Public Const Pi = 3.14159265
```

HUOMAA Tarkempia tietoja vakiomoduuleista on harjoituksessa 9.

Seuraava esimerkki havainnollistaa vakion käyttämistä tapahtuma-aliohjelmassa.

Vakion käyttö tapahtuma-aliohjelmassa

1. Valitse File-valikosta komento Open Project.

   Open Project -valintaikkuna avautuu.
2. Avaa Constant-projekti kansiosta \Vb5Sbs\Less04.
3. Napsauta Project-ikkunan Näytä objekti -painiketta.

   Constant-ohjelman lomake tulee näkyviin. Tämäkin on tynkäohjelma; käyttöliittymä on valmis, mutta sinun pitää lisätä ohjelmalauseet.
4. Kaksoisnapsauta lomakkeen Näytä vakio -painiketta.

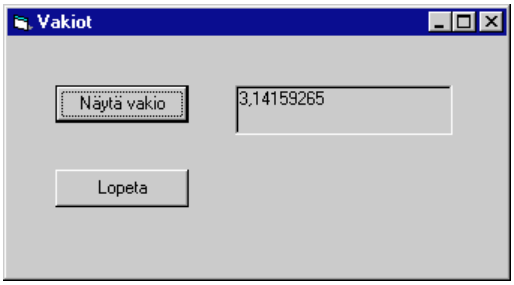
   Tapahtuma-aliohjelma Command1_Click ilmestyy Code-ikkunaan.
5. Kirjoita aliohjelmaan seuraavat lauseet:

   ```
   Const Pi = 3.14159265
   Label1.Caption = Pi
   ```
6. Käynnistä ohjelma napsauttamalla työkalurivin Käynnistä-painiketta.

Käynnistä-painike

7 Napsauta ohjelman Näytä vakio -painiketta.

Vakion Pi arvo näkyy tekstiruudussa tällaisena:



8 Lopeta ohjelma napsauttamalla Lopeta-painiketta.

Jos haluat säilyttää kopion muokatusta Constant-ohjelmasta, tallenna lomake ja projekti nimellä **MyConstant**.

Vakiot ovat käteviä varsinkin matemaattisissa kaavoissa, esimerkiksi Ala = $2\pi r^2$. Seuraavassa jaksossa kuvataan, miten muodostat operaattoreista ja muuttujista tämänkaltaisia lausekkeita.

Visual Basicin operaattorien käyttö

Visual Basicin operaattoreilla muodostetaan kaavoja.

Lauseke on lukujen, muuttujien, operaattorien ja avainsanojen yhdistelmä, josta saadaan arvo. Visual Basic -kielessä on monenlaisia osia, joista muodostetaan lausekkeita. Tässä jaksossa pääset kokeilemaan matemaattisia operaattoreita eli symboleita, joilla kootaan yhteen matemaattisen lausekkeen osat. Melkein kaikki Visual Basicin matemaattiset symbolit ovat sinulle tuttuja tavanomaisista laskutehtävistä, ja niiden merkitys on kutakuinkin selvä. Esittelen kaikki matemaattiset operaattorit seuraavilla sivuilla.

Visual Basicissa on seuraavat operaattorit:

Operaattori	Toiminta
+	Yhteenlasku
–	Vähennyslasku
*	Kertolasku
/	Jakolasku
\	Kokonaislukujen jakolasku
Mod	Jakojäännös
^	Potenssiin korotus
&	Merkkijonojen yhteenliittäminen

Alkeismatematiikkaa: operaattorit +, –, * ja /

Yhteen-, vähennys-, kerto- ja jakolaskuoperaattorit eivät juuri tuota vaikeuksia, ja niitä voidaan käyttää missä tahansa lausekkeessa, joka sisältää lukuja tai lukuarvoisia muuttujia. Seuraavaksi esiteltävä ohjelma kuvaa näiden operaattorien soveltamista ohjelmassa.

Perusoperaattorien käyttö

1. Valitse File-valikosta komento Open Project.
2. Avaa Basic-Op-projekti kansiosta \Vb5Sbs\Less04.
3. Valitse lomake Project-ikkunasta ja napsauta Näytä objekti -painiketta.

   Perusoperaattorit-ohjelman lomake tulee näkyviin. Sillä voit kokeilla, miten yhteen-, vähennys-, kerto- ja jakolaskuoperaattorit käsittelevät näppäimillä syöttämiäsi lukuja. Siitä näet myös, miten tekstiruutu-, valintanappi- ja komentopainikeobjekteilla luetaan syöttötietoja.
4. Napsauta työkalurivin Käynnistä-painiketta.

   Perusoperaattorit-ohjelma käynnistyy ohjelmointiympäristön alaisuudessa. Se näyttää kaksi tekstiruutua, joihin voit kirjoittaa lukuja, joukon valintanappeja, tekstikehyksen, joka näyttää tuloksen, ja kaksi komento-painiketta.

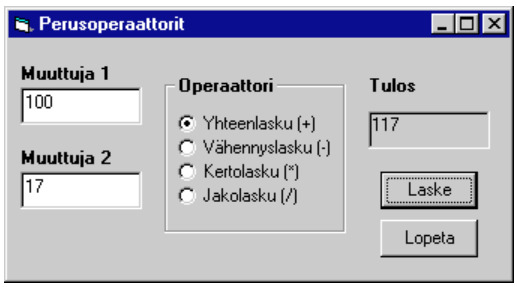
Tekstiruutuobjekti on kätevä keino lukea syöttö-tietoja näppäi-mistöstä.

5. Kirjoita Muuttuja 1 -tekstiruutuun **100** ja paina SARKAINNÄPPÄINTÄ.

   Kohdistin siirtyy toiseen tekstiruutuun.
6. Kirjoita Muuttuja 2 -tekstiruutuun 17.

   Nyt voit soveltaa tekstiruuduissa oleviin lukuihin mitä tahansa neljästä matemaattisesta operaatiosta.
7. Napsauta Yhteenlasku-nappia ja napsauta Laske-painiketta.

   Ohjelma laskee arvot yhteen ja näyttää Tulos-ruudussa tuloksen eli luvun 117. Lomakkeen pitäisi näyttää tällaiselta:



8 Laske muuttujaruuduissa olevien lukujen erotus, tulo ja osamäärä valitsemalla sopivat operaattorit. Napsauta aina operaattorin valittuasi Laske-painiketta.

Tulos näkyy Tulos-ruudussa. Voit kirjoittaa tekstiruutuihin muitakin lukuja. (Kokeile myös lukua, jossa on desimaalipiste ja desimaaliosa.)

9 Kun olet lopettanut laskemisen, napsauta Lopeta-painiketta.

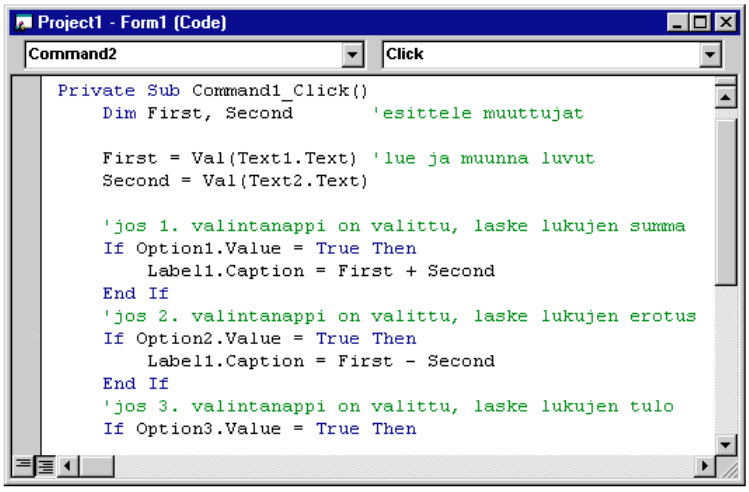
Ohjelman suoritus päättyy, ja ohjelmointiympäristö palaa näkyviin.

Katso seuraavaksi ohjelmalauseista, miten tulokset lasketaan. Peruoperaattorit-ohjelma perustuu muutamaan syöttöobjektiin, joihin tutustuit harjoituksessa 2, ja tapahtuma-aliohjelmaan, joka laskee yksinkertaisia matemaattisia lausekkeita muuttujilla ja operaattoreilla. Aliohjelma myös muuttaa tekstiruudusta lukemansa merkkijonon luvuksi Val-funktiolla.

Perusoperaattorit-ohjelman tutkiminen

1 Kaksoisnapsauta lomakkeen Laske-painiketta.

Tapahtuma-aliohjelma Command1_Click ilmestyy Code-ikkunaan:



Aliohjelman ensimmäinen lause esittelee kaksi yleiskäyttöistä, muunnelmatyyppistä muuttujaa. Näihin sijoitetaan luvut, jotka käyttäjä kirjoittaa tekstiruutuihin. Ne ovat niin joustavia, että niihin voidaan tallentaa mikä tahansa lukuarvo. Kaksi seuraavaa lausetta muuntavat tekstiruuduista luetut merkkijonot luvuiksi Val-funktiolla ja sijoittavat tulokset muuttujiin:

```
First = Val(Text1.Text) 'lue ja muunna luvut
Second = Val(Text2.Text)
```

Val-funktio muuntaa tekstin luvuksi.

Val-funktio on erikoisaliohjelma, joka muuntaa tekstiargumentin luvuksi. Muuntaminen on tarpeen, jotta yhteenlaskuoperaatio toimisi oikein tässä ohjelmassa. Tekstiruutuobjektin palauttaman tiedon oletustyyppi on teksti. Tämä ei ole ongelma ohjelman kolmen operaattorin yhteydessä. Operaattorit –, * ja / käsittelevät vain lukuja, joten jos käyttäjä valitsee yhden näistä, Visual Basic muuntaa tekstiruuduista lukemansa merkkijonot automaattisesti luvuiksi.

Sen sijaan yhteenlaskuoperaattori + koskee sekä merkkijonoja että lukuja. Koska tekstiruutuobjekti palauttaa normaalisti tekstin eli merkkijonon, Visual Basic tulkitsisi käyttäjän antamat luvut merkkijonoiksi. Se liittäisi merkkijonot peräkkäin sen sijaan että se muuntaisi ne luvuiksi ja laskisi tulokset yhteen. Esimerkiksi ”100” + ”17” tuottaisi tuloksen ”10017”.

Seuraavassa tehtävässä opit lisää merkkijonojen liittämisestä peräkkäin. Paina tässä vaiheessa mieleesi, että vaikka muunnelmatyyppi voi vastata mitä tahansa perustietotyyppiä, sinun on tarkasti seurattava, miten muuttujan arvoa käytetään ohjelmassa. Ohjelma saattaa toimia eri tavoin kuin luulet.

TÄRKEÄÄ On aina tärkeää testata ohjelman jokainen laskutoimitus ohjelman oikean toiminnan varmistamiseksi. Ohjelman yhden osan testaaminen ei riitä.

2 Vieritä Code-ikkunaa alaspäin ja tarkastele neljää kaavaa, joissa on matemaattisia perusoperaattoreita.

Aliohjelman ensimmäinen osa sisältää yhteenlaskuoperaattorin (+) If...Then-rakenteessa:

```
'jos 1. valintanappi on valittu, laske lukujen summa
If Option1.Value = True Then
    Label1.Caption = First + Second
End If
```

Jos ensimmäisen valintanapin Value-ominaisuus on True eli valintanappia on napsautettu, kaksi muuttujaa lasketaan yhteen +-operaattorilla, ja tulos sijoitetaan tekstiobjektin ominaisuuteen. Kolme muuta lauseketta toimivat samalla tavoin: jokainen käyttää If...Then-rakennetta ja Label1-objektin Caption-ominaisuutta. Ehtorakenteet, kuten If...Then, ovat olennaisen tärkeitä, kun ohjelmassa pitää valita suoritettavat lauseet käyttäjän tekemän valinnan perusteella. Opit lisää If...Then-rakenteesta seuraavassa harjoituksessa.

3 Sulje Code-ikkuna.

Olemme nyt käsitelleet Perusoperaattorit-ohjelman loppuun.

Visual Basicin matemaattiset funktiot

Ohjelmissa tarvitaan silloin tällöin hieman numeronmurskausta. Sinun pitää ehkä muuntaa arvo toisentyyppiseksi, laskea monimutkainen matemaattinen lauseke tai lisätä ohjelmaan satunnaislukuja. Alla luetellut Visual Basicin funktiot helpottavat lukujen käsittelemistä. Muiden funktioiden tapaan matemaattisia funktioita käytetään ohjelmalauseissa, ja ne palauttavat arvon. Seuraavassa taulukossa argumentti *n* edustaa lukua, muuttujaa tai lauseketta, jota funktion on tarkoitus käsitellä.

Funktio	Kuvaus
Abs(*n*)	Palauttaa luvun *n* itseisarvon.
Atn(*n*)	Palauttaa luvun *n* arkustangentin, joka on ilmaistu radiaaneina.
Cos(*n*)	Palauttaa kulman *n* kosinin. Kulma on ilmaistu radiaaneina.
Exp(*n*)	Palauttaa vakion *e* korotettuna potenssiin *n*.
Rnd(*n*)	Tuottaa satunnaisluvun 0–1.
Sgn(*n*)	Palauttaa arvon −1, jos *n* on pienempi kuin 0, arvon 0, jos *n* on 0, tai arvon +1, jos *n* on suurempi kuin 0.
Sin(*n*)	Palauttaa kulman *n* sinin. Kulma on ilmaistu radiaaneina.
Sqr(*n*)	Palauttaa luvun *n* neliön eli toisen potenssin.
Str(*n*)	Muuntaa luvun merkkijonoksi.
Tan(*n*)	Palauttaa kulman *n* tangentin. Kulma on ilmaistu radiaaneina.
Val(*n*)	Muuntaa merkkijonon luvuksi.

Kehittyneemmät operaattorit: \, Mod, ^ ja &

Visual Basicissa on neljän matemaattisen perusoperaattorin lisäksi neljä kehittyneempää operaattoria: kokonaislukujen jakolasku (\), jakojäännös (Mod), potenssiin korotus (^) ja merkkijonojen yhteenliittäminen (&). Niistä on hyötyä erikoisemmissa matemaattisissa lausekkeissa ja tekstinkäsittelyssä. Seuraava ohjelma (joka on johdettu Perusoperaattorit-ohjelmasta) kuvaa näiden operaattoreiden käyttöä.

Kehittyneempien operaattorien käyttö

1. Valitse File-valikosta komento Open Project.
2. Avaa AdvOp-projekti kansiosta \Vb5Sbs\Less04.

3 Valitse lomake Project-ikkunasta ja napsauta Näytä objekti -painiketta.

Kehittyneet operaattorit -ohjelman lomake tulee näkyviin. Se on samanlainen kuin Perusoperaattorit-ohjelmassa, mutta valintanapeissa on eri operaattorit. Ohjelman sisäiset operaattorit ovat myös erilaiset.

4 Napsauta työkalurivin Käynnistä-painiketta.

Ohjelma näyttää kaksi tekstiruutua lukuarvojen kirjoittamista varten, muutamia operaattorien valintanappeja, tekstiruudun tulosta varten ja kaksi komentopainiketta.

5 Kirjoita Muuttuja 1 -tekstiruutuun 9 ja napsauta SARKAINNÄPPÄINTÄ.

6 Kirjoita Muuttuja 2 -tekstiruutuun 2.

Nyt voit soveltaa tekstiruuduissa oleviin arvoihin yhtä neljästä kehittyneemmästä operaattorista.

7 Napsauta Kokonaislukujako-nappia ja sitten Laske-painiketta.

Ohjelma käsittelee arvot operaattorilla ja näyttää Tulos-ruudussa luvun 4, kuten seuraavasta kuvasta näkyy:

Kokonaislukujen jakolaskun tulos sisältää vain jakolaskun kokonaisosan. Vaikka 9 jaettuna 2:lla on 4.5, operaattori palauttaa vain tämän luvun kokonaisosan eli 4. Tällaisesta operaattorista on hyötyä, kun on kyse kokonaisuuksista, joita ei voi helposti jakaa pienempiin osiin, esimerkiksi ihmisistä.

8 Napsauta Jakojäännös-nappia ja sitten Laske-painiketta.

Tulos-ruudussa näkyy luku 1. Jakojäännösoperaattori (modulus) palauttaa osamäärän jakojäännöksen eli jaettavasta jääneen osan, kun jakolasku ei mene tasan. Koska 9 jaettuna 2:lla on 4 ja jakojäännös on 1 ($2 \times 4 + 1 = 9$), Mod-operaattorilla saadaan tulos 1.

9 Napsauta Potenssi-nappia ja sitten Laske-painiketta.

Tulos-ruudussa näkyy luku 81. Potenssiinkorotusoperaattori (^) kertoo luvun itsellään niin monta kertaa kuin halutaan. Visual Basicissa 9 ^ 2 tarkoittaa samaa kuin 9^2, ja tämähän on 81.

10 Napsauta Yhteenliittäminen-nappia ja sitten Laske-painiketta.

Tulos-ruudussa näkyy merkkijono ”92”. Merkkijonojen yhteenliittämis-operaattori (&) yhdistää kaksi merkkijonoa. Tulos, tässä tapauksessa 92 , ei ole luku, vaan merkkien ”9” ja ”2” yhdistelmä. Voit liittää peräkkäin ainoastaan merkkijonomuuttujia, lainausmerkein ympäröityjä merkkijonoja ja muunnelmamuuttujia. Koska tämän ohjelman muuttujat ovat muunnelmatyyppisiä, ne muuttuvat automaattisesti merkkijonoiksi yhteenliittämistä varten. Jos haluat kokeilla, miten yhteenliittäminen toimii merkkijonoilla, kirjoita kumpaankin muuttujaruutuun pari sanaa ja napsauta Laske-painiketta uudelleen.

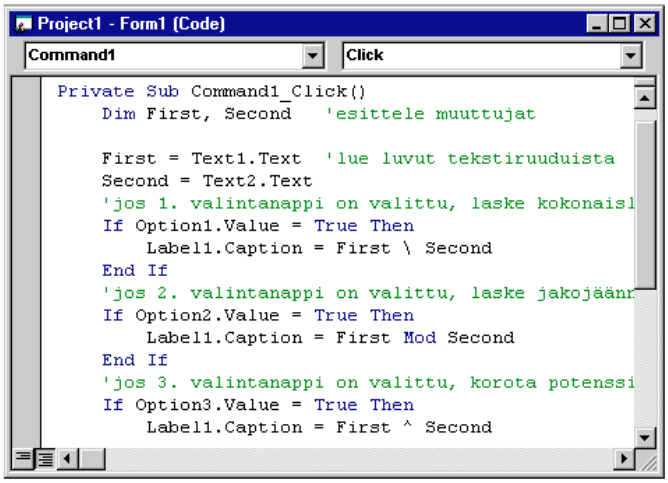
11 Lopeta ohjelma napsauttamalla Lopeta-painiketta.

Ohjelman suoritus päättyy, ja ohjelmointiympäristö palaa näkyviin.

Tutki hieman tapahtuma-aliohjelmaa Command1_Click, jotta näet, miten siinä käytetään operaattoreita.

12 Kaksoisnapsauta lomakkeen Laske-painiketta.

Tapahtuma-aliohjelma ilmestyy Code-ikkunaan seuraavan kuvan mukaisena:



Tämä Command1_Click-aliohjelma muistuttaa Perusoperaattorit-ohjelman samannimistä aliohjelmaa. Se esittelee kaksi muunnelmamuuttujaa, sijoittaa niihin tekstiruuduista luetut tiedot ja laskee valitun kaavan If...Then-rakenteella.

Aliohjelmassa on kuitenkin yksi tärkeä ero: siinä ei muunneta teksti-ruuduista luettuja tietoja luvuiksi Val-funktiolla. Muuntaminen ei ole tarpeen tässä kuvattujen operaattorien yhteydessä, koska jokainen niistä, yhteenlaskuoperaattorista (+) poiketen, käsittelee vain yhdentyyppisiä tietoja. \, Mod ja ^ koskevat ainoastaan lukuja ja & ainoastaan tekstiä.

Koska tyyppien moniselitteisyyttä ei esiinny, muunnelmamuuttujat pystyvät ongelmitta muuntamaan tekstiruutuobjekteista saadut merkkijonot luvuiksi niitä operaattoreita varten, jotka käsittelevät vain lukuja.

13 Sulje Code-ikkuna.

Kehittyneet operaattorit -ohjelmakin on nyt loppuun käsitelty.

Operaattorien laskentajärjestys

Kahdessa edellisessä tehtävässä kokeilit seitsemää matemaattista operaattoria ja yhtä merkkijono-operaattoria. Visual Basic -lauseeseen saa kirjoittaa miten monta matemaattista operaattoria tahansa, kunhan jokainen lukuarvoinen muuttuja tai lauseke on erotettu toisesta yhdellä operaattorilla. Esimerkiksi tämä on sallittu Visual Basic -lause:

```
Summa = 10 + 15 * 2 / 4 ^ 2
```

Lause suorittaa useita operaatioita ja sijoittaa tuloksen muuttujaan nimeltä Summa. Miten Visual Basic laskee tällaisen lausekkeen? Toisin sanoen, mitkä operaatiot se suorittaa ensin? Olet varmaan jo huomannut, että laskentajärjestys vaikuttaa suuresti tällaisen lausekkeen tulokseen.

Operaattorien laskentajärjestys on syytä pitää mielessä matemaattisia lausekkeita kirjoitettaessa.

Visual Basicissa ongelma on ratkaistu määrittämällä matemaattisille operaattoreille *laskentajärjestys* eli prioriteetti. Se muodostuu säännöistä, joiden perusteella Visual Basic valitsee laskutoimitusten suoritusjärjestyksen, kun lausekkeessa on useita operaattoreita. Seuraavassa taulukossa on esitetty laskentajärjestys ensimmäisestä operaattorista viimeiseen. Jos kaksi operaattoria on laskentajärjestyksessä samalla tasolla, ne lasketaan vasemmalta oikealle.

Operaattori	Laskentajärjestys
( )	Sulkeilla ympäröidyt operaatiot lasketaan aina ensin.
^	Potenssiin korotus lasketaan toiseksi.
–	Negaatio (etumerkin vaihto) lasketaan kolmanneksi.
* /	Kerto- ja jakolasku ovat neljäntenä.
\	Kokonaislukujen jakolasku on viidentenä.
Mod	Jakojäännös on kuudentena.
+ –	Yhteen- ja vähennyslasku ovat viimeisinä.

Taulukossa esitettyjen laskentajärjestyssääntöjen mukaan Visual Basic laskee lausekkeen

```
Summa = 10 + 15 * 2 / 4 ^ 2
```

seuraavissa vaiheissa (kulloinkin suoritettava operaatio on lihavoitu).

```
Summa = 10 + 15 * 2 / 4 ^ 2
Summa = 10 + 15 * 2 / 16
Summa = 10 + 30 / 16
Summa = 10 + 1.875
Summa = 11.875
```

Lisätietoja: Sulkeiden käyttö kaavassa

Sulkeet selventävät laskentajärjestystä ja vaikuttavat siihen.

Voit käyttää lausekkeessa yhtä tai useampia suljepareja laskentajärjestyksen selventämiseksi. Jos esimerkiksi kirjoitat lauseen

```
Luku  = (8 - 5 * 3) ^ 2
```

Visual Basic laskee sulkeissa olevan lausekkeen arvon (–7) ja vasta sitten potenssiinkorotuksen, vaikka potenssiinkorotus muuten laskettaisiin ennen vähennys- ja kertolaskua. Voit määrätä laskentajärjestyksen vielä tarkemmin käyttämällä sisäkkäisiä sulkeita. Esimerkiksi lauseen

```
Luku  = ((8 - 5) * 3) ^ 2
```

sulkeet määräävät, että Visual Basic laskee ensin sisempien sulkeiden sisällä olevan erotuksen, sitten kertolaskun ja lopuksi potenssiinkorotuksen. Näiden lausekkeiden tulokset ovat erilaiset: ensimmäisestä saadaan 49 ja toisesta 81. Sulkeet siis sekä tekevät lausekkeesta helpomman lukea että vaikuttavat sen tulokseen.

Jos haluat jatkaa seuraavaan harjoitukseen

➤ Jätä Visual Basic käyntiin ja siirry harjoitukseen 5.

Jos haluat lopettaa Visual Basicin

➤ Valitse File-valikosta komento Exit.

Jos näyttöön tulee Save-valintaikkuna, napsauta Yes.

Harjoituksen tiivistelmä

Tehtävä	Ohje

Esitellä muuttuja	Kirjoita **Dim** ja muuttujan nimi. Voit myös määrittää tietotyypin; muunnelmatyyppi on oletus. Esimerkki:
	`Dim Apu% 'Kokonaislukutyyppi`
	`Dim Muisti 'Muunnelmatyyppi`

Vaihtaa muuttujan arvo	Sijoita muuttujaan uusi arvo sijoitusoperaattorilla (=). Esimerkki: `Maa = "Japani"`
Lukea syöttötietoja valintaikkunasta	Käytä InputBox-funktiota ja sijoita tulos muuttujaan. Esimerkki: `Kayttaja = InputBox("Mikä on nimesi?")`

Näyttää tuloksia valintaikkunassa	Käytä MsgBox-funktiota. Valintaikkunassa näytettävä merkkijono voidaan sijoittaa muuttujaan. Esimerkki:
	`Ennuste = "Sataa vettä, räntää tai lunta."`
	`MsgBox(Ennuste), , "Yleinen sääennuste"`

Esitellä tietyntyyppinen muuttuja	Kirjoita **Dim**, muuttujan nimi ja tyyppimääritysmerkki.
	tai
	Kirjoita **Dim**, muuttujan nimi, avainsana **As** ja perustietotyypin nimi. (Perustietotyyppejä on kahdeksan.) Esimerkki:
	`Dim SyntPaiva As Date 'Päivämäärä`
	`Dim Hinta! 'yksinkert. tarkkuuden liukuluku`

Luoda vakio	Kirjoita avainsana **Const**, vakion nimi, sijoitusoperaattori (=) ja vakion arvo. Esimerkki: `Const MatinIka = 39`
Kirjoittaa kaava	Liitä toisiinsa lukuarvoisia muuttujia tai arvoja käyttäen mitä tahansa seitsemästä matemaattisesta operaattorista ja sijoita tulos muuttujaan tai ominaisuuteen. Esimerkki: `Tulos = 1 ^ 2 * 3 \ 4 'tämä on 0`
Yhdistää merkkijonoja	Käytä merkkijonojen yhteenliittämisoperaattoria (&). Esimerkki: `Viesti = "Huomenta" & "," & " maailma!"`

Tehtävä	Ohje
Muuntaa merkkijono luvuksi	Käytä Val-funktiota. Esimerkki: `Pii = Val("3.1415926535897932")`
Käyttää matemaattista funktiota	Kirjoita kaavaan funktio ja sen mahdolliset argumentit. Esimerkki: `Hypotenuusa = Sqr(x ^ 2 + y ^ 2)`

Määrätä kaavan laskentajärjestys	Käytä kaavassa sulkeita. Esimerkkejä:
	`Tulos = 1 + 2 ^ 3 \ 4  'tämä on 3`
	`Tulos = (1 + 2) ^ ( 3 \ 4)  'tämä on 1`

Kun tarvitset tietoja aiheesta	Valitse Visual Basicin Help-valikosta Microsoft Visual Basic Help Topics, valitse Index-välilehti ja...
Muuttujien esitteleminen	Kirjoita **variables, declaring**.
Perustietotyypit	Kirjoita sen tietotyypin symboli, josta haluat tietoja.
Käyttäjän määrittämät tietotyypit	Kirjoita **data types, user-defined**.
InputBox-funktion käyttö	Kirjoita **InputBox**.
MsgBox-funktion käyttö	Kirjoita **MsgBox**.
Operaattorit ja laskentajärjestys	Kirjoita operaattori, josta haluat tietoja.
Matemaattiset funktiot	Kirjoita sen funktion nimi, josta haluat tietoja.

Tietoja seuraavasta harjoituksesta

Seuraavassa harjoituksessa opit ohjaamaan tapahtuma-aliohjelman lauseiden suoritusta. Näytän, miten If...Then- ja Select Case -rakenteita käytetään ja miten ohjelmasta suoritetaan eri lohkoja sen mukaan, mitkä ehdot toteutuvat. Kerron myös, miten voit etsiä ohjelmasta kielioppi- ja loogisia virheitä ja korjata ne.

Harjoitus

5

Ehtorakenteiden käyttö

Arvioitu aika
45 min.

Tässä harjoituksessa opit

- kirjoittamaan ehtolauseita
- valitsemaan suoritettavat ohjelmalauseet ehdon perusteella If...Then-rakenteella
- valitsemaan yhden useasta vaihtoehdosta Select Case -rakenteella
- etsimään ja korjaamaan ohjelman virheitä.

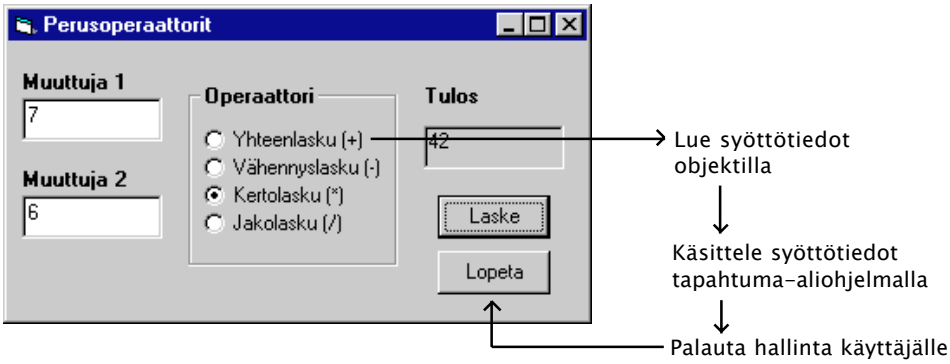
Edellisissä harjoituksissa käsittelit käyttäjän antamat tiedot Visual Basicin eri työkaluilla. Näytit vaihtoehtoja valikoissa, objekteissa ja valintaikkunoissa ja käsittelit syöttötiedot ominaisuuksilla ja muuttujilla. Tässä harjoituksessa näet, miten tietty osa ohjelmasta valitaan suoritettavaksi käyttäjältä saadun tiedon perusteella. Opit tutkimaan yhden tai useita ominaisuuksia tai muuttujia ehtolausekkeilla ja sitten valitsemaan suoritettavat lauseet saatujen tulosten mukaan. Opit myös havaitsemaan ja korjaamaan ohjelmointivirheitä käyttämällä askellustilaa.

Tapahtumaohjattu ohjelmointi

Visual Basic -ohjelmat ovat tapahtuma-ohjattuja.

Tähän mennessä kirjoittamasi Visual Basic -ohjelmat ovat näyttäneet valikoita, objekteja ja valintaikkunoita ja kehottaneet käyttäjää käsittelemään näyttöä sopivaksi katsomallaan tavalla. Ohjelma antaa vuoron käyttäjälle, odottaa kärsivällisesti vastausta ja käsittelee annetut tiedot ennustettavalla tavalla.

Ohjelmoijien keskuudessa tällaista menettelyä sanotaan *tapahtumaohjatuksi ohjelmoinniksi*. Ohjelma muodostetaan tekemällä joukko ”älykkäitä” objekteja, jotka osaavat reagoida käyttäjän toimintoihin. Syöttötiedot käsitellään käyttämällä objekteihin kytkettyjä tapahtuma-aliohjelmia. Seuraava kaavio esittää, miten tapahtumaohjattu ohjelma toimii Visual Basicissa:

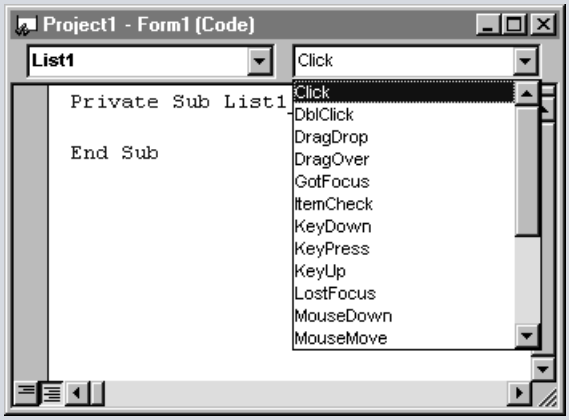


Ohjelman syöttötiedot voivat olla peräisin myös itse tietokonejärjestelmästä. Ohjelma voi esimerkiksi saada tiedon siitä, että sähköpostiviesti on saapunut tai tietty aika on kulunut järjestelmäkellon mukaan. Nämä tapahtumat aiheuttaa tietokone eikä käyttäjä. Tapahtuman syystä riippumatta Visual Basic reagoi kutsumalla siihen objektiin liittyvää tapahtuma-aliohjelmaa, joka havaitsi tapahtuman. Vaikka olet tähän mennessä kokeillut vain napsautus- ja muuttumistapahtumia (Click ja Change), Visual Basic -objektit osaavat reagoida moniin muihinkin tapahtumiin.

Visual Basicin tapahtumaohjattu luonne merkitsee sitä, että suurin osa tehtävistä suoritetaan tapahtuma-aliohjelmissa. Nämä tapahtumakohtaiset aliohjelmat käsittelevät syöttötietoja, laskevat uusia arvoja, näyttävät tuloksia ja hoitavat muita tehtäviä. Edellisessä harjoituksessa opit suorittamaan tapahtuma-aliohjelmassa laskutoimituksia käyttämällä muuttujia, operaattoreita ja matemaattisia lausekkeita. Tässä harjoituksessa opit vertaamaan muuttujia, ominaisuuksia ja arvoja ehtorakenteilla sekä valitsemaan suoritettavat lauseet vertailun tulosten perusteella. Seuraavassa harjoituksessa muodostat silmukoita, jotka toistavat lauseryhmää yhä uudelleen, kunnes annettu ehto toteutuu. Näistä tehokkaista ohjausrakenteista tehdään aliohjelmia, jotka pystyvät reagoimaan melkein mihin tahansa tilanteeseen.

Visual Basicin objekteihin liittyvät tapahtumat

Jokaisella Visual Basicin objektilla on valmiiksi määritetty joukko tapahtumia, joihin se reagoi. Nämä tapahtumat on lueteltu kunkin objektin avattavassa Proc (Procedure) -luetteloruudussa, joka näkyy Code-ikkunassa. Voit kirjoittaa tapahtuma-aliohjelman jokaista tapahtumaa varten, ja jos kyseinen tapahtuma sattuu, Visual Basic käynnistää vastaavan tapahtuma-aliohjelman. Esimerkiksi luetteloruutuobjekti tuntee tapahtumat Click, DblClick, DragDrop, DragOver, GotFocus, ItemCheck, KeyDown, KeyPress, KeyUp, LostFocus, MouseDown, MouseMove, MouseUp, OLECompleteDrag, OLEDragDrop, OLEDragOver, OLEGiveFeedback, OLESetData, OLEStartDrag ja Scroll. Sinun tuskin tarvitsee varautua useampaan kuin yhteen tai kahteen tapahtumaan ohjelmassasi, mutta on hyvä tietää, että käyttöliittymän osia luotaessa on näin paljon valinnanvaraa. Seuraavassa kuvassa näkyy osa luetteloruutuobjektin tapahtumista:



Ehtolausekkeiden käyttö

Ehtolausekkeet muodostavat totuusarvoisia kysymyksiä.

Yksi hyödyllisimmistä välineistä käsitellä tietoja tapahtuma-aliohjelmassa on ehtolauseke. *Ehtolauseke* on ohjelmalauseen osa, joka muodostaa totuusarvoisen kysymyksen koskien ominaisuutta, muuttujaa tai muuta ohjelman tietoa. Esimerkiksi ehtolauseke

```
Hinta < 100
```

palauttaa totuusarvon True (tosi), jos muuttujan Hinta arvo on pienempi kuin 100, tai totuusarvon False (epätosi), jos muuttujan Hinta arvo on yhtä suuri tai suurempi kuin 100. Voit käyttää ehtolausekkeissa seuraavalla sivulla esitettyjä vertailuoperaattoreita.

Vertailu-operaattori	Merkitys
=	Yhtä suuri kuin
< >	Erisuuri kuin
>	Suurempi kuin
<	Pienempi kuin
>=	Suurempi tai yhtä suuri kuin
<=	Pienempi tai yhtä suuri kuin

HUOMAA Lauseketta, jonka tulos on True tai False, sanotaan myös loogiseksi lausekkeeksi. Tulos on mahdollista tallentaa loogiseen muuttujaan tai ominaisuuteen. Voit sijoittaa loogisia arvoja tiettyihin objektin ominaisuuksiin, muunnelmamuuttujiin tai loogisiin muuttujiin, jotka on esitelty Dim-lauseella käyttäen avainsanoja As Boolean.

Alla olevassa taulukossa on muutamia ehtolausekkeita ja niiden tuloksia. Seuraavassa tehtävässä voit kokeilla taulukossa lueteltuja operaattoreita.

Ehtolauseke	Tulos
10 < > 20	True (10 ei ole yhtä suuri kuin 20).
Pisteet < 20	True, jos Pisteet on alle 20, muuten False.
Pisteet = Label1.Caption	True, jos Label1-objektin Caption-ominaisuus sisältää saman arvon kuin Pisteet-muuttuja, muuten False.
Text1.Text = "Ville"	True, jos ensimmäisessä tekstiruudussa on nimi *Ville*, muuten False.

If...Then-ehtorakenteet

If...Then-ehtorakenteilla lisätään ohjelmaan logiikkaa.

Ehtolausekkeilla ohjataan ohjelman lauseiden suoritusjärjestystä, kun niitä käytetään ehtorakenteeksi sanotussa lauselohkossa. If...Then-ehtorakenteella tutkitaan ehto ja valitaan toimintatapa tuloksen perusteella. Yksinkertaisimmassa muodossaan If...Then-ehtorakenne kirjoitetaan yhdelle riville:

```
If ehto Then lause
```

Ehto on ehtolauseke ja *lause* Visual Basicin sääntöjen mukainen ohjelmalause. Esimerkiksi

```
If Pisteet >= 20 Then Label1.Caption = "Sinä voitit!"
```

on If...Then-ehtorakenne, jossa ehtolausekkeella

```
Pisteet >= 20
```

päätetään, pitääkö ohjelman sijoittaa Label1-objektin Caption-ominaisuuteen teksti ”Sinä voitit!”. Jos Pisteet-muuttujan arvo on suurempi tai yhtä suuri kuin 20, Visual Basic määrittää Caption-ominaisuuden. Muuten se ohittaa sijoituslauseen ja siirtyy tapahtuma-aliohjelman seuraavaan lauseeseen. Tällaisen vertailun tulos on aina tosi tai epätosi, ei koskaan ”ehkä”.

Usean ehdon tutkiminen If...Then-ehtorakenteessa

Visual Basicissa on myös sellainen If...Then-ehtorakenne, johon voidaan sisällyttää useita ehtolausekkeita. Tällainen lauselohko voi olla monta riviä pitkä ja sisältää tärkeät avainsanat ElseIf, Else ja End If.

ElseIf- ja Else-osia käyttäen laaditaan lisäkysymyksiä If...Then-rakenteeseen.

```
If ehto1 Then
    lauseet, jotka suoritetaan, jos ehto1 on tosi
ElseIf ehto2 Then
    lauseet, jotka suoritetaan, jos ehto2 on tosi
[tähän voidaan lisätä muita ElseIf-osia ja lauseita]
Else
    lauseet, jotka suoritetaan, jos mikään ehdoista ei ole tosi
End If
```

Tässä rakenteessa *ehto1* lasketaan ensin. Jos se on tosi, sitä seuraava lauselohko suoritetaan (siinä voi olla yksi tai useita lauseita). Ellei ensimmäinen ehto ole tosi, toinen ehtolauseke (*ehto2*) lasketaan. Jos tämä on tosi, toinen lauselohko suoritetaan. Jos tutkittavia ehtoja on enemmän, voit kirjoittaa lisää ElseIf-lauseita ja ehtoja. Ellei mikään ehtolausekkeista ole tosi, Else-avainsanaa seuraavat lauseet suoritetaan. Koko rakenne päätetään avainsanoilla End If.

Seuraava monirivinen If...Then-rakenne sopisi tuloveron laskemiseen. Tulorajat ja veroprosentit ovat vuoden 1996 veroasteikon mukaiset.

Moniriviset If...Then-rakenteet sopivat erinomaisesti tilanteeseen, jossa pitää selvittää, mille alueelle jokin arvo osuu.

```
If Tulot <= 43000 Then          ' vakiovero
    Vero = 50
ElseIf Tulot <= 59000 Then      ' vero 7 %
    Vero = 50 + ((Tulot - 43000) * 0.7)
ElseIf Tulot <= 73000 Then      ' vero 17 %
    Vero = 1170 + ((Tulot - 59000) * 0.17)
ElseIf Tulot <= 104000 Then     ' vero 21 %
    Vero = 3550 + ((Tulot - 73000) * 0.21)
ElseIf Tulot <= 163000 Then     ' vero 27 %
    Vero = 10060 + ((Tulot - 104000) * 0.27)
ElseIf Tulot <= 290000 Then     ' vero 33 %
    Vero = 25990 + ((Tulot - 163000) * 0.33)
Else                            ' vero 39 %
    Vero = 67900 + ((Tulot - 290000) * 0.39)
End If
```

Tässä ehtorakenteessa Tulot-muuttujaa verrataan ensimmäiseen tulorajaan ja sitten seuraaviin tulorajoihin, kunnes jokin ehtolausekkeista palauttaa arvon True, ja veronmaksajan tulovero voidaan laskea. Tämä yksinkertainen ehtorakenne on hyvin käyttökelpoinen. Sillä voidaan laskea kenen tahansa verot progressiivisessa verojärjestelmässä, kunhan veroprosentit ovat ajan tasalla ja Tulot-muuttujan arvo on oikea. Jos veroprosentit muuttuvat, ehtolausekkeita on helppo muuttaa.

TÄRKEÄÄ If...Then- ja Elseif-osien ehtolausekkeiden järjestys on erittäin tärkeä. Jos kirjoitat veronlaskentaesimerkin ehtolausekkeet päinvastaiseen järjestykseen (suurimmasta tulorajasta pienimpään), pienituloisetkin veronmaksajat sijoittuvat korkeinta progressiivista veroprosenttia vastaavaan luokkaan, koska he ovat ansainneet enintään 290 000 mk. (Visual Basic valitsee ensimmäisen toden ehtolausekkeen, vaikka muutkin olisivat tosia.) Koska tämän esimerkin kaikki ehtolausekkeet tutkivat saman muuttujan, ne on kirjoitettava nousevaan järjestykseen, jotta vero laskettaisiin oikein. Opetus: Kun kirjoitat useita ehtolausekkeita, mieti tarkoin, missä järjestyksessä niiden tulee olla.

Seuraavassa esimerkissä käytät If...Then-ehtorakennetta tarkistaessasi, onko käyttäjällä ohjelman käyttöoikeus. Voisit soveltaa samankaltaista logiikkaa tehdessäsi verkkosovellusta.

Käyttöoikeuksien tarkistaminen If...Then-lauseella

1. Käynnistä Visual Basic.

   Jos Visual Basic on jo käynnissä, avaa uusi projekti.

2. Napsauta Komentopainike-työkalua ja piirrä lomakkeen vasempaan yläkulmaan komentopainike.

Komentopainike-työkalu

3. Kirjoita komentopainikkeen Caption-ominaisuudeksi ”Sisäänkirjaudu”.
4. Kaksoisnapsauta Sisäänkirjaudu-painiketta.

   Tapahtuma-aliohjelma Command1_Click ilmestyy Code-ikkunaan.

If...Then-, ElseIf- ja Else-avainsanoja seuraavat lauseet on tapana sisentää.

5 Kirjoita aliohjelmaan seuraavat lauseet:

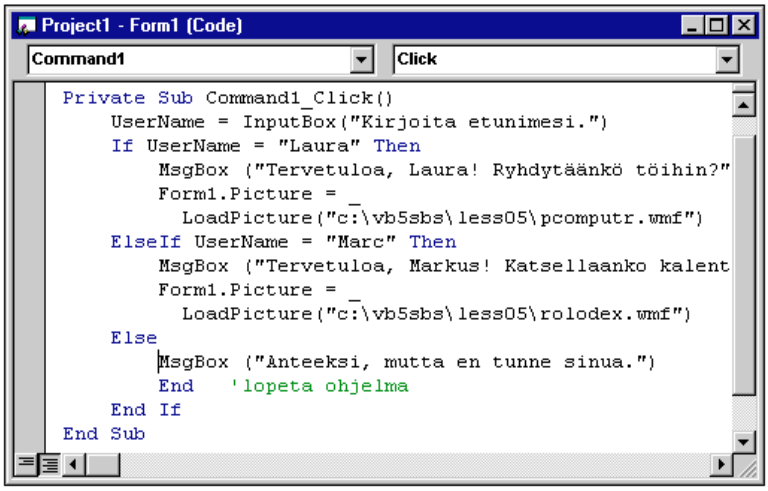
```
UserName = InputBox("Kirjoita etunimesi.")
If UserName = "Laura" Then
     MsgBox("Tervetuloa, Laura! Ryhdytäänkö töihin?")
     Form1.Picture = _
        LoadPicture("c:\vb5sbs\less05\pcomputr.wmf")
ElseIf UserName = "Markus" Then
     MsgBox("Tervetuloa, Markus! Katsellaanko kalenteria?")
     Form1.Picture = _
        LoadPicture("c:\vb5sbs\less05\rolodex.wmf")
Else
     MsgBox("Anteeksi, mutta en tunne sinua.")
     End   'lopeta ohjelma
End If
```

Kummankin Form1.Picture-ominaisuuden jäljessä on rivinjatkomerkki (_), jolla kaksi pitkää ohjelmalausetta jaetaan kahtia, jotta ne mahtuvat kirjan sivulle. Voit halutessasi kirjoittaa kummankin lauseen yhdelle riville; Code-ikkuna vierittyy tarvittaessa oikealle.

HUOMAA Visual Basicin Code-ikkunaan voidaan kirjoittaa ohjelmarivi, jossa on peräti 1023 merkkiä, mutta yleensä riville kannattaa kirjoittaa enintään 80 merkkiä. Voit jakaa pitkän lauseen usealle riville lisäämällä rivinjatkomerkin (_) jokaisen muun rivin paitsi viimeisen loppuun. (Et kuitenkaan voi jakaa lainausmerkein ympäröityä merkkijonoa kahdelle riville.)

Kun olet valmis, Project-ikkunan pitäisi näyttää tällaiselta:

Valmis Login-ohjelma on kansiossa \Vb5Sbs\Less05.



Käynnistä-painike

6 Napsauta työkalurivin Käynnistä-painiketta.

Ohjelma käynnistyy ohjelmointiympäristön alaisuudessa. Näyttöön tulee tyhjä lomake, jonka vasemmassa yläkulmassa on Sisäänkirjaudu-painike.

7 Napsauta Sisäänkirjaudu-painiketta.

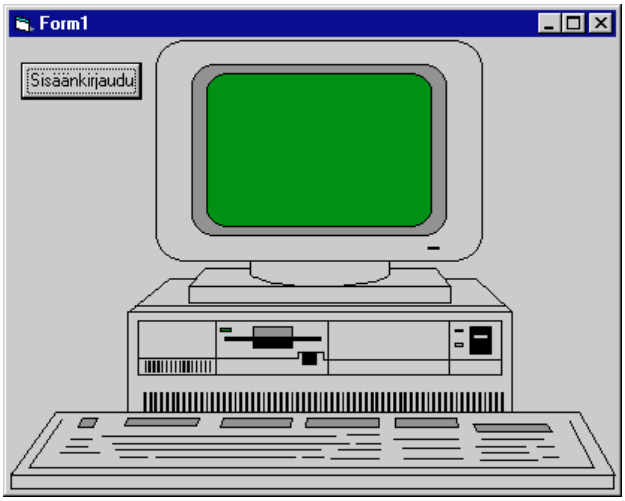
Tapahtuma-aliohjelma Command1_Click kutsuu Input-funktiota, joka kysyy etunimesi valintaikkunassa.

8 Kirjoita **Laura** ja paina ENTER.

If…Then-ehtorakenteen ensimmäinen ehtolauseke vertaa kirjoittamaasi nimeä merkkijonoon "Laura". Jos kirjoitit *Laura*, lasuekkeen arvo on True, ja ohjelma näyttää tervetulotoivotuksen kutsumalla MsgBox-funktiota.

9 Napsauta ilmoitusikkunassa OK.

Ohjelma sulkee ilmoitusikkunan ja lataa lomakkeeseen Windows-metatiedoston, jossa on mikrotietokoneen kuva. Katso seuraavaa kuvaa.



Tämä ohjelma lataa Windows-metatiedoston suoraan lomakkeeseen käyttämällä Picture-ominaisuutta. Lomakkeella on samanlainen Picture-ominaisuus kuin kuva- ja kuvakehysobjekteilla. Kun kuva on ladattu lomakkeeseen, se näkyy taustalla. Lomakkeeseen lisätyt objektit ovat kuvan päällä.

10 Napsauta Sisäänkirjaudu-painiketta, kirjoita **Markus** ja napsauta OK.

Tällä kertaa ehtorakenne valitsee ElseIf-osan ja päästää Markus-nimisen käyttäjän ohjelmaan. Ohjelma näyttää jälleen tervetulotoivotuksen kutsumalla MsgBox-funktiota.

11 Näytä kortiston kuva napsauttamalla OK.

Ohjelma lataa lomakkeeseen Windows-metatiedoston.

12 Napsauta Sisäänkirjaudu-painiketta, kirjoita **Romppanen** ja napsauta OK.

Nyt suoritetaan ehtorakenteen Else-osa, joka näyttää seuraavan ilmoituksen MsgBox-objektissa:

13 Sulje ilmoitusikkuna napsauttamalla OK.

Ilmoitusikkuna katoaa, ja ohjelman suoritus päättyy. Valtuuttamattomalla henkilöllä ei ole ohjelman käyttöoikeutta.

14 Tallenna lomake nimellä **MyLogin.frm** ja projekti nimellä **MyLogin.vbp**.

Loogisten operaattorien käyttö ehtolausekkeissa

Loogisilla operaattoreilla lisätään lausekkeisiin monio-saisia testejä.

Visual Basicin If...Then- ja ElseIf-lauseisiin voidaan kirjoittaa useita ehtolausekkeita, jos ehtorakenteessa tarvitaan enemmän kuin yksi valintaehto. Ehdot liitetään toisiinsa seuraavilla loogisilla operaattoreilla:

Looginen operaattori	Kuvaus
And (ja)	Jos kumpikin ehtolauseke on tosi, tulos on tosi.
Or (tai)	Jos ainakin toinen ehtolauseke on tosi, tulos on tosi.
Not (ei)	Jos ehtolauseke on epätosi, tulos on tosi. Jos ehtolauseke on tosi, tulos on epätosi.
Xor (joko-tai)	Jos ainoastaan toinen ehtolauseke on tosi, tulos on tosi. Jos kumpikin ehtolauseke on tosi tai kumpikin on epätosi, tulos on epätosi.

HUOMAA Kun ohjelma laskee monimutkaista lauseketta, jossa on erityyppisiä operaattoreita, se suorittaa ensin matemaattiset operaatiot, sitten vertailuoperaatiot ja lopuksi loogiset operaatiot.

Seuraavassa taulukossa on esimerkkejä loogisista operaattoreista. Siinä oletetaan, että muuttuja Ajoneuvo sisältää merkkijonon ”Pyörä” ja muuttuja Hinta sisältää arvon 2000.

Looginen lauseke	Tulos
Vehicle = ”Pyörä” And Hinta < 3000	Tosi (kumpikin ehto on tosi)
Vehicle = ”Auto” Or Hinta < 5000	Tosi (toinen ehto on tosi)
Not Hinta < 1000	Tosi (ehto on epätosi)
Vehicle = ”Pyörä” Xor Hinta < 3000	Epätosi (kumpikin ehto on tosi)

Seuraava tehtäväsi on muokata MyLogin-ohjelmaa niin, että se kysyy käyttäjältä myös salasanan. Kysy salasana valintaikkunassa ja muokkaa ehtorakenteen If...Then- ja ElseIf-lauseita niin, että ne tarkistavat salasanan käyttäen And-operaattoria.

Salasanasuojauksen lisääminen And-operaattorilla

1. Avaa tapahtuma-aliohjelma Command1_Click Code-ikkunaan.
2. Lisää seuraava lause InputBox-lauseen ja If…Then-lauseen väliin (ensimmäisen ja toisen rivin väliin):

   ```
   Pass = InputBox("Kirjoita salasanasi.")
   ```

3. Muuta If...Then-lause tällaiseksi:

   ```
   If UserName = "Laura" And Pass = "Touko17" Then
   ```

   Lauseessa on nyt looginen And-operaattori, joka tarkistaa käyttäjätunnuksen ja salasanan ennen kuin Laura saa luvan käyttää ohjelmaa.

Valmis Password-ohjelma on kansiossa \Vb5Sbs\Less05.

4. Muuta ElseIf-lause tällaiseksi:

   ```
   ElseIf UserName = "Markus" And Pass = "trek" Then
   ```

5. Tallenna lomake nimellä **MyPass.frm** ja projekti nimellä **MyPass.vbp**.
6. Napsauta työkalurivin Käynnistä-painiketta.

   Ohjelma käynnistyy ohjelmointiympäristön alaisuudessa.

Käynnistä-painike

7. Napsauta Sisäänkirjaudu-painiketta, kirjoita **Laura** ja napsauta OK.

   Ohjelma kysyy salasanan.
8. Kirjoita **Touko17** ja napsauta OK.

   And-operaattorin sisältävä ehtolauseke saa arvokseen True, ja ohjelma toivottaa Lauran tervetulleeksi.
9. Sulje ilmoitusikkuna napsauttamalla OK.

Lopeta-painike

10 Lopeta ohjelma napsauttamalla työkalurivin Lopeta-painiketta.

Ohjelman suoritus päättyy, ja ohjelmointiympäristö palaa näkyviin.

HUOMAA Jos aiot tehdä MyPass-ohjelmasta laajemman version, harkitse salasanan kysymistä tekstiruutuobjektilla. Tekstiruutuobjektissa on PasswordChar-ominaisuus, jolla voidaan korvata käyttäjän kirjoittamat merkit esimerkiksi tähdellä (*), ja MaxLength-ominaisuus, jolla rajoitetaan kirjoitettavien merkkien määrä.

Select Case -ehtorakenteet

Select Case-ehtorakenne perustuu yhteen avainmuuttujaan.

Visual Basic -ohjelman lauseiden suoritusta voidaan ohjata myös Select Case -ehtorakenteilla. Käytit niitä jo aikaisemmin tässä kirjassa, kun kirjoitit tapahtuma-aliohjelmia luetteloruuduista ja valikoista valittujen toimintojen käsittelemistä varten. Select Case -rakenne muistuttaa If..Then...ElseIf-rakennetta, mutta on tehokkaampi, kun haarautuminen riippuu vain yhdestä muuttujasta. Se myös tekee ohjelmasta helpommin ymmärrettävän ja tehokkaamman.

Select Case -rakenne näyttää tällaiselta:

```
Select Case muuttuja
Case arvo1
    lauseet, jotka suoritetaan, jos muuttuja on yhtä kuin arvo1
Case arvo2
    lauseet, jotka suoritetaan, jos muuttuja on yhtä kuin arvo2
Case arvo3
    lauseet, jotka suoritetaan, jos muuttuja on yhtä kuin arvo3
.
.
.
End Select
```

Select Case -rakenne alkaa avainsanoilla Select Case ja päättyy avainsanoihin End Select. Kirjoita *muuttujan* tilalle muuttuja, ominaisuus tai muu lauseke, joka toimii rakenteen avainarvona. Korvaa *arvo1*, *arvo2* ja *arvo3* luvuilla, merkkijonoilla tai muilla arvoilla, jotka ovat samaa tyyppiä kuin tutkittava muuttuja. Jos jokin näistä arvoista on yhtä suuri kuin muuttuja, Visual Basic suorittaa kyseisen arvon jäljessä olevat lauseet ja siirtyy sitten End Select -lausetta seuraavaan lauseeseen. Voit kirjoittaa Select Case -rakenteeseen miten monta Case-osaa tahansa, ja yhdessä Case-osassa saa olla useita arvoja. Peräkkäiset arvot on eroteltava pilkuin.

Seuraavasta esimerkistä näkyy, miten Select Case -lauseella voitaisiin näyttää erilaisia ilmoituksia henkilön iän mukaan. Jos Ika-muuttuja on yhtä suuri kuin annettu Case-arvo, ohjelma näyttää tekstiruudussa sopivan ilmoituksen.

```
Select Case Ika
Case 16
     Label1.Caption = "Saat kevarikortin!"
Case 18
     Label1.Caption = "Voit mennä naimisiin!"
Case 21
     Label1.Caption = "Olet virallisesti täysi-ikäinen."
Case 65
     Label1.Caption = "Aika siirtyä eläkkeelle ja pitää hauskaa!"
End Select
```

Select Case -rakenne voi olla muodoltaan selkeämpi kuin vastaava If... Then-rakenne.

Select Case -rakenteessa voi olla myös Case Else -osa, joka suoritetaan, ellei mikään edellä luetelluista ehdoista toteudu. Sitä voitaisiin soveltaa edelliseen esimerkkiin näin:

```
Select Case Ika
Case 16
     Label1.Caption = "Saat kevarikortin!"
Case 18
     Label1.Caption = "Voit mennä naimisiin!"
Case 21
     Label1.Caption = "Olet virallisesti täysi-ikäinen."
Case 65
     Label1.Caption = "Aika siirtyä eläkkeelle ja pitää hauskaa!"
Case Else
     Label1.Caption = "Olet parhaassa iässä. Nauti siitä!"
End Select
```

Vertailuoperaattorien käyttö Select Case -rakenteessa

Select Case -rakenteessa voi olla samanlaisia vertailuoperaattoreita kuin If...Then-rakenteessa.

Voit määrittää Select Case -rakenteessa tarkkojen arvojen lisäksi arvoalueita. Tätä varten ovat käytettävissä Visual Basicin vertailuoperaattorit =, < >, >, <, >= ja <=. Jos Case-osan lausekkeessa on jokin näistä operaattoreista, siinä tarvitaan myös avainsana Is tai To. Kääntäjä vertaa testimuuttujaa lausekkeeseen, joka on Is-avainsanan jäljessä. To-avainsana taas määrittää arvoalueen. Seuraava rakenne käyttää Is- ja To-avainsanoja sekä useita vertailuoperaattoreita, kun se tutkii Ika-muuttujan ja näyttää yhden viidestä ilmoituksesta.

```
Select Case Ika
```

```
Case Is < 13
     Label1.Caption = "Nuoruuden kulta-aikaa!"
Case 13 To 19
     Label1.Caption = "Nauti teini-iästä!"
Case 21
     Label1.Caption = "Olet virallisesti täysi-ikäinen."
Case Is > 100
     Label1.Caption = "Hyvältä näyttää!"
Case Else
     Label1.Caption = "Olet ihan mukavassa iässä."
End Select
```

Jos Ika-muuttujan arvo on alle 13, ohjelma näyttää ilmoituksen ”Nuoruuden kulta-aikaa!” Jos Ika-muuttuja on 13–19, ohjelma näyttää ilmoituksen ”Nauti teini-iästä!” ja niin edelleen.

Select Case -rakenne on tavallisesti paljon selvempi kuin If...Then-rakenne ja tehokkaampi, kun valitaan yksi kolmesta tai useammasta vaihtoehdosta yhden muuttujan tai ominaisuuden perusteella. Jos vaihtoehtoja on vain kaksi tai jos tutkittavia arvoja on useita, If...Then-rakenne on todennäköisesti suositeltavampi.

Seuraavassa ohjelmassa näet, miten Select Case -rakenteella käsitellään luetteloruudusta valittu vaihtoehto. Lue syöttötieto käyttämällä ominaisuuksia List1.Text ja LIst1.listIndex ja näytä sen perusteella yksi neljästä erikielisestä tervehdyksestä.

Luetteloruudun käsittely Select Case -rakenteella

1. Valitse File-valikosta komento New Project ja luo vakiosovellus.

   Ohjelmointiympäristössä näkyy tyhjä lomake.

Teksti-työkalu

2. Napsauta Teksti-työkalua ja piirrä lomakkeen yläosan keskelle suuri kehys, johon tulee ohjelman otsikko.

Luetteloruutu-työkalu

3. Napsauta Luetteloruutu-työkalua ja piirrä otsikon alapuolelle luetteloruutu.

Komentopainike-työkalu

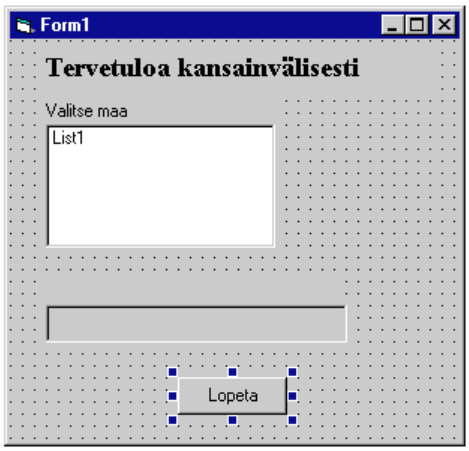
4. Lisää luetteloruutuobjektin yläpuolelle pieni tekstiobjekti ja luetteloruudun alapuolelle kaksi pientä tekstiobjektia ohjelman tulosten näyttämistä varten.

Properties-ikkuna-painike

5. Napsauta Komentopainike-työkalua ja piirrä pieni komentopainike lomakkeen alaosaan.
6. Napsauta työkalurivin Properties-ikkuna-painiketta ja määritä objektien ominaisuudet seuraavan sivun taulukon mukaisesti.

Objekti	Ominaisuus	Asetus
Label1	Caption	”Tervetuloa kansainvälisesti”
	Font	Times New Roman, Bold, 14 pistettä
Label2	Caption	”Valitse maa”
Label3	Caption	(Tyhjä)
Label4	Caption	(Tyhjä)
	BorderStyle	1 - Fixed Single
	ForeColor	Dark red (&H00000080&)
Command1	Caption	”Lopeta”

Kun olet määrittänyt objektien ominaisuudet, lomakkeen pitäisi näyttää tällaiselta:



Kirjoita ohjelmalauseet, jotka alustavat luetteloruudun.

7 Kaksoisnapsauta lomaketta.

Tapahtuma-aliohjelma Form_Load ilmestyy Code-ikkunaan.

8 Kirjoita seuraavat ohjelmalauseet, joilla alustetaan luetteloruutu:

Luetteloruutuun lisätään vaihtoehtoja AddItem-metodilla.

```
List1.AddItem  "Englanti"
List1.AddItem  "Saksa"
List1.AddItem  "Espanja"
List1.AddItem  "Suomi"
```

Nämä lauseet lisäävät vaihtoehdot lomakkeen luetteloruutuun kutsumalla AddItem-metodia.

9 Avaa objektiluettelo ja valitse siitä objekti List1.

Tapahtuma-aliohjelma List1_Click ilmestyy Code-ikkunaan.

10 Kirjoita seuraavat lauseet, jotka käsittelevät käyttäjän valitseman vaihtoehdon:

ListIndex-ominaisuus sisältää luettelosta valitun vaihtoehdon numeron.

```
Label3.Caption = List1.Text
Select Case List1.ListIndex
Case 0
     Label4.Caption = "Hello, programmer"
Case 1
     Label4.Caption = "Hallo, programmierer"
Case 2
     Label4.Caption = "Hola, programador"
Case 3
     Label4.Caption = "Hei, ohjelmoija"
End Select
```

Ensimmäinen lause kopioi luetteloruudusta valitun vaihtoehdon nimen lomakkeen kolmanteen tekstiruutuun. Lauseen tärkein ominaisuus on List1.Text, joka sisältää valitun vaihtoehdon tarkan tekstin. Seuraavat lauseet kuuluvat Select Case -rakenteeseen. Ne käyttävät ominaisuutta LIst1.ListIndex testimuuttujana ja vertaavat sitä useisiin arvoihin. ListIndex sisältää aina luetteloruudusta valitun vaihtoehdon numeron: ensimmäinen vaihtoehto on numero 0, toinen numero 1, kolmas numero 2 ja niin edelleen. ListIndex-ominaisuuden ansiosta Select Case -rakenne voi nopeasti selvittää, minkä vaihtoehdon käyttäjä valitsi, ja toimia oikealla tavalla.

11 Avaa objektiluettelo ja valitse siitä objekti Command1.

Tapahtuma-aliohjelma Command1_Click ilmestyy Code-ikkunaan.

12 Kirjoita aliohjelmaan **End** ja sulje Code-ikkuna.

13 Tallenna lomake nimellä **MyCase.frm** ja projekti nimellä **MyCase.vbp**.

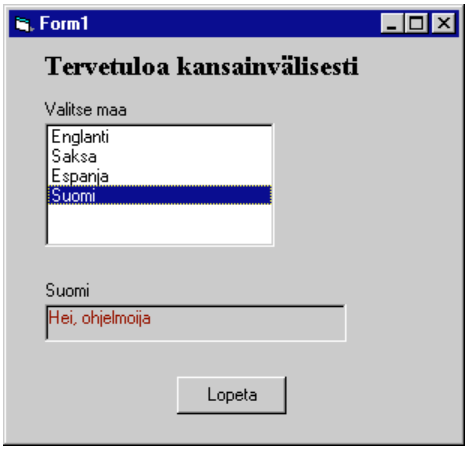
14 Käynnistä MyCase-ohjelma napsauttamalla työkalurivin Käynnistä-painiketta.

Käynnistä-painike

15 Valitse maiden nimiä Valitse maa -luetteloruudusta.

Ohjelma näyttää tervehdyksen kielellä, jota valitsemassasi maassa puhutaan. Seuraavan sivun kuvassa on suomenkielinen teksti.

Valmis Case-ohjelma on kansiossa \Vb5Sbs\Less05.



16 Lopeta ohjelma napsauttamalla Lopeta-painiketta.

Ohjelman suoritus päättyy, ja ohjelmointiympäristö palaa näkyviin. Lopetamme Select Case -rakenteiden käsittelyn tähän.

Virheiden etsiminen ja korjaaminen

Ohjelman virheiden etsimistä ja korjaamista sanotaan testaamiseksi.

Tähän mennessä havaitsemasi ohjelmointivirheet ovat varmaankin olleet yksinkertaisia kirjoitus- tai kielioppivirheitä. Entä jos huomaat ohjelmassa hankalamman ongelman, jota et pysty löytämään ja korjaamaan vain tutkimalla ohjelman objekteja, ominaisuuksia ja lauseita? Visual Basic -ohjelmointiympäristössä on useita välineitä, joilla jäjitetään ja korjataan ohjelmien virheitä. Ne eivät estä sinua tekemästä virheitä, mutta niistä on usein apua, kun olet todennut ohjelman toimivan väärin.

Katso seuraavaa If...Then-rakennetta, joka laskee kaksi ehtolauseketta ja näyttää tuloksen perusteella toisen kahdesta ilmoituksesta:

```
If Ika > 13 AND Ika < 20 Then
     Text2.Text = "Olet teini-iässä."
Else
     Text2.Text = "Et ole teini-iässä."
End If
```

Huomaatko virheen? Teini-ikäinen tarkoittaa henkilöä, jonka ikä on 13–19 vuotta. Näiden lauseiden mukaan tasan 13-vuotias ei ole teini-ikäinen. Kyseessä ei ole kielioppivirhe, sillä lauseet ovat Visual Basicin sääntöjen mukaiset, vaan ajatus- eli *looginen virhe*. Oikeassa ehtorakenteessa olisi suurempi tai yhtä suuri kuin -operaattori (>=):

```
If Ika >= 13 AND Ika < 20 Then
```

Usko tai älä, mutta tällaiset virheet ovat kaikkein yleisimpiä. Ohjelma, joka toimii oikein suurimman osan ajasta, mutta ei aina, on kaikkein työläin testata ja korjata.

Kolmenlaisia virheitä

Visual Basic -ohjelmassa voi ilmetä kolmenlaisia virheitä: kielioppi-, ajonaikaisia ja loogisia virheitä.

- *Kielioppivirhe* (eli *kääntämisenaikainen virhe*) on ohjelmointivirhe, joka rikkoo Visual Basicin sääntöjä, esimerkiksi väärin kirjoitettu ominaisuuden nimi tai avainsana. Visual Basic pystyy osoittamaan monenlaisia kielioppivirheitä jo ohjelmalauseita kirjoittaessasi. Se ei anna sinun käynnistää ohjelmaa ennen kuin jokainen kielioppivirhe on korjattu.
- *Ajonaikainen virhe* on virhe, joka aiheuttaa ohjelman suorituksen keskeytymisen. Se johtuu esimerkiksi ulkoisesta tapahtumasta tai havaitsematta jääneestä kielioppivirheestä. Väärin kirjoitettu tiedoston nimi LoadPicture-funktiossa tai auki jätetty levykeaseman salpa ovat tilanteita, jotka voivat johtaa ajonaikaiseen virheeseen.
- *Looginen virhe* on inhimillinen erehdys eli ohjelmointivirhe, joka saa ohjelman tuottamaan vääriä tuloksia. Ohjelman testaamisessa on useimmiten kyse ohjelmoijan loogisten virheiden etsimisestä.

Lue Visual Basicin ohjeita, jos näyttöön tulee kielioppi- tai ajonaikaisesta virheestä johtuva ilmoitus. Napsauta ilmoitusikkunan Help-painiketta.

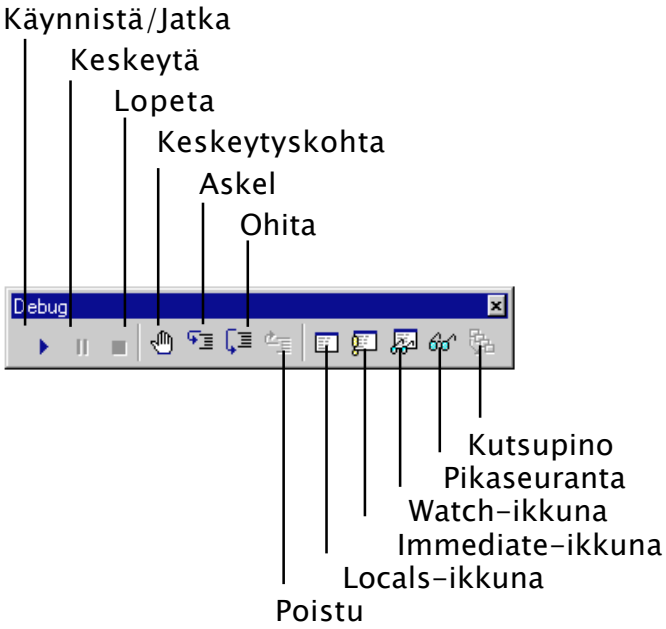
Askellustilan käyttö

Askellustilassa voit nähdä, miten ohjelman suoritus etenee.

Yksi tapa löytää looginen virhe on suorittaa ohjelma lause kerrallaan ja tutkia yhden tai useamman muuttujan tai ominaisuuden arvon muuttumista. Tätä varten sinun pitää valita *askellustila* ohjelman suorituksen aikana ja tarkkailla lauseiden suoritusta Code-ikkunassa. Askellustilassa saat tarkan kuvan ohjelmasta, kun Visual Basic -kääntäjä suorittaa sitä. Voit ikään kuin seurata, miten kapteeni ja perämies ohjaavat lentokonetta, ja tarvittaessa jopa tarttua ohjaimiin.

Kun testaat ohjelmaa, sinun kannattaa tuoda esiin Debug-työkalurivi, jossa on virheiden jäljittämiseen tarkoitettuja painikkeita. Avaa myös Watches-ikkuna, josta näet kiinnostavien ja tärkeiden muuttujien sisällöt. Kolmas mahdollisuus on Immediate-ikkuna, johon voit kirjoittaa ohjelmalauseita ja nähdä niiden vaikutuksen heti.

Seuraavassa kuvassa näkyy Debug-työkalurivi, jonka voit avata valitsemalla View-valikosta komennon Toolbars ja valitsemalla työkalurivien luettelosta vaihtoehdon Debug.



Seuraavassa tehtävässä käytät askellustilaa, kun etsit ja korjaat loogisen virheen, jonka havaitsit edellä esitetyssä If...Then-rakenteessa. (Virhe on kirjoitettu yhteen tämän kirjan valmiista malliohjelmista.) Yritä paikantaa ongelma siten, että suoritat ohjelmalauseita yksi kerrallaan napsauttamalla Debug-työkalurivin Askel-painiketta ja seuraat Age-muuttujan arvon muuttumista napsauttamalla Pikaseuranta-painiketta. Kiinnitä erityistä huomiota tähän testausmenetelmään. Sillä voit löytää ja korjata monenlaisia virheitä omista ohjelmistasi.

Avaa projekti -painike

IfBug-ohjelman testaaminen

1. Napsauta työkalurivin Avaa projekti -painiketta.
2. Avaa IfBug-projekti kansiosta \Vb5Sbs\Less05.
3. Ellei lomaketta näy, valitse IfBug-lomake Project-ikkunasta ja napsauta Näytä objekti -painiketta.

   If-virhe-lomake tulee esiin. Ohjelma kysyy käyttäjältä iän. Kun käyttäjä painaa Testaa-painiketta, ohjelma ilmoittaa, onko hän teini-ikäinen. Jos käyttäjä on 13-vuotias, ohjelma tuottaa väärän tuloksen, kuten aiemmin tässä harjoituksessa näimme. Tuo Debug-työkalurivi näkyviin nyt ja etsi virhe askellustilassa.

4 Valitse View-valikosta komento Toolbars ja napsauta vaihtoehtoa Debug, ellei sitä ole vielä valittu.

Debug-työkalurivi tulee näkyviin. Se on ehkä telakoitu Standard-työkalurivin oikealle puolelle.

5 Vedä Debug-työkalurivi If-virhe-lomakkeen alapuolelle, jotta se on helposti käsillä testatessasi ohjelmaa.

Käynnistä-painike

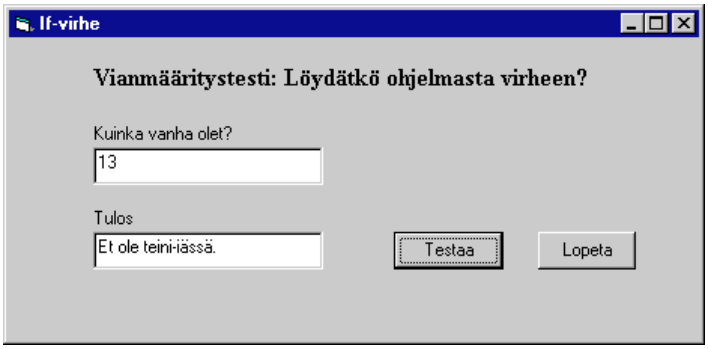
6 Napsauta Debug-työkalurivin Käynnistä-painiketta.

7 Ohjelma käynnistyy. Poista Ikä-tekstiruudusta numero 0, kirjoita sen tilalle 14 ja napsauta Testaa-painiketta.

Ohjelma ilmoittaa, että olet teini-ikäinen. Tällä kertaa tulos on siis oikea.

8 Kirjoita Age-tekstiruutuun 13 ja napsauta Testaa-painiketta.

Ohjelma ilmoittaa, että et ole teini-ikäinen, kuten seuraavasta kuvasta näkyy.

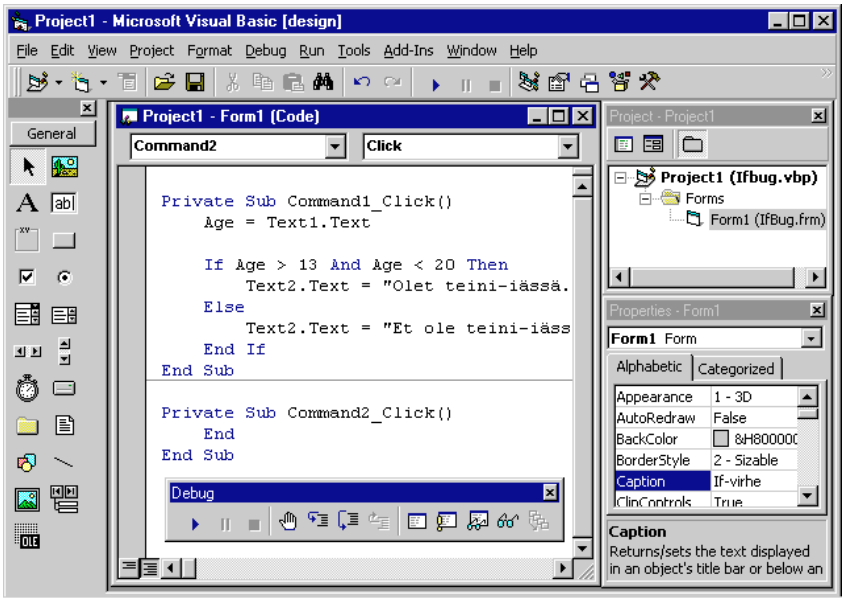


Vastaus on väärä, ja sinun pitää tutkia ohjelmalauseita löytääksesi ongelman. Sen sijaan, että lopettaisit ohjelman ja tutkisit lauseita omin päin, anna Visual Basicin auttaa.

9 Napsauta Debug-työkalurivin Keskeytä-painiketta (Käynnistä-painikkeen oikealla puolella).

Keskeytä-painike

Ohjelman suoritus keskeytyy, ja Visual Basic näyttää Code-ikkunan, joka esittää ohjelmalauseet suorituksen aikana. Näytön pitäisi olla seuraavan sivun kuvan mukainen.



Askel-painike

10 Suorita seuraava ohjelmalause napsauttamalla Debug-työkalurivin Askel-painiketta.

Visual Basic luovuttaa hallinnan lomakkeelle ja jää odottamaan syöttötietoja.

11 Napsauta Windowsin tehtäväpalkissa näkyvää IfBug-painiketta, varmista, että tekstiruudussa lukee edelleen 13, ja napsauta Testaa-painiketta.

Koska Visual Basic on askellustilassa, jotakin epätavallista tapahtuu. Visual Basic avaa Code-ikkunan ja näyttää tapahtuma-aliohjelman Command1_Click eli ohjelmalauseet, jotka kääntäjän on tarkoitus suorittaa. Aliohjelman ensimmäinen lause on korostettu keltaisella. Nyt voit tarkkailla, miten ohjelman logiikka toimii.

12 Suorita aliohjelman ensimmäinen lause napsauttamalla Askel-painiketta.

Visual Basic suorittaa Sub-lauseen ja korostaa Age-muuttujan sisältävän lauseen. Age on kriittinen testimuuttuja tässä ohjelmassa, joten sinun kannattaa lisätä se Watches-ikkunaan ja seurata sen muuttumista ohjelman suorituksen aikana.

VIHJE Kun ohjelma on askellustilassa, voit tarkistaa Code-ikkunassa näkyvän muuttujan arvon osoittamalla muuttujaa ja pitämällä hiiren painiketta alhaalla.

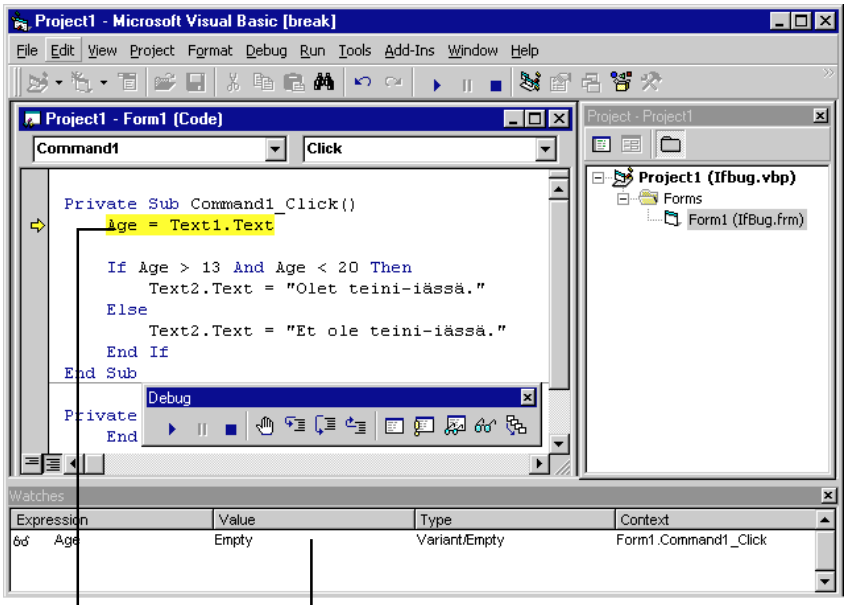
Pikaseuranta-painike

13 Valitse Age-muuttuja hiirellä ja napsauta Debug-työkalurivin Pikaseuranta-painiketta.

Näyttöön avautuu valintaikkuna, joka näyttää Age-muuttujan kontekstin, nimen ja arvon. Näet tämän valintaikkunan myös valittuasi Debug-valikosta komennon Quick Watch.

Watches-ikkuna sisältää muuttujat, jotka on lisätty Pikaseuranta-komennolla.

14 Napsauta Quick Watch -valintaikkunan Add-painiketta. Watches-ikkuna avautuu telakoituna näytön alareunaan. (Sinun pitää ehkä muuttaa sen kokoa nähdäksesi sen kokonaan.)



Seuraavaksi suoritettava lause

Age-muuttuja Watches-ikkunassa

Age-muuttujalla ei ole nyt arvoa, koska sitä ei ole vielä käytetty tapahtuma-aliohjelmassa. Age-muuttujaa ei ole esitelty ohjelmassa julkiseksi, joten se on aliohjelman paikallinen muuttuja ja alustetaan uudelleen aina aliohjelmaa kutsuttaessa.

HUOMAA Jos haluat poistaa seurattavan muuttujan Watches-ikkunasta, napsauta sitä Watches-ikkunassa ja paina DEL.

15 Suorita seuraava lause napsauttamalla Askel-painiketta.

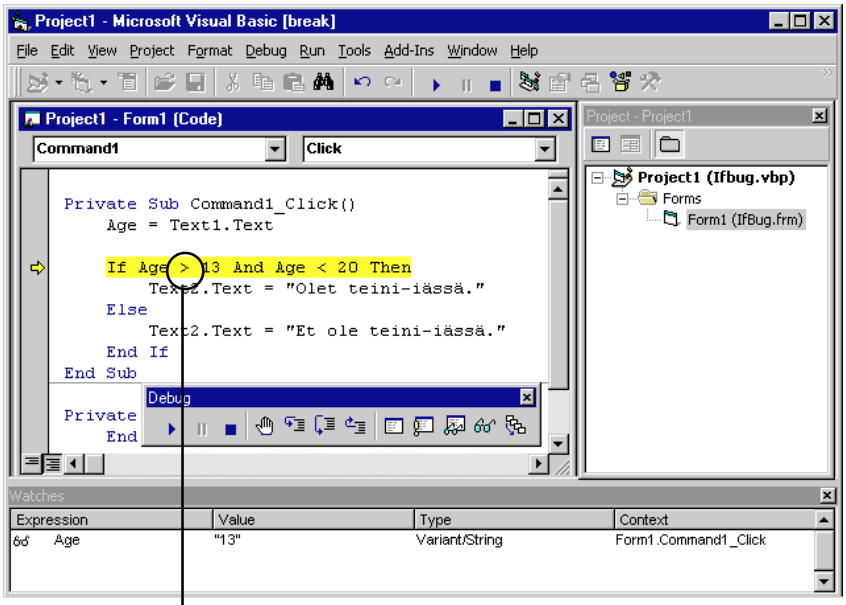
Jos ehtolause aiheuttaa siirtymisen väärään paikkaan, etsi virhettä ehtolausekkeesta.

Visual Basic kopioi luvun 13 tekstiruudusta Age-muuttujaan ja päivittää Watches-ikkunassa näkyvän Age-muuttujan arvon. Sitten Visual Basic korostaa If...Then-rakenteen ensimmäisen lauseen eli ohjelman tärkeimmän (ja virheellisen) lauseen. Koska 13-vuotias nuori on teini-ikäinen, Visual Basicin pitäisi siirtyä tämän lauseen jälkeen Then-lauseeseen.

16 Napsauta taas Askel-painiketta.

Visual Basic korostaakin If...Then-rakenteen Else-osan. Tällä arvolla testin tulos on väärä, ja sinun pitää löytää ja korjata virhe. Visual Basic kyllä auttoi löytämään virheellisen lauseen, mutta sinun on paikannettava ja korjattava virhe itse. Tällä kertaa tiedät jo vastauksen: ensimmäisessä vertailulausekkeessa pitää olla operaattori >=.

17 Napsauta operaattorin > oikeaa puolta If...Then-lauseessa ja kirjoita =. Ikkunan pitäisi näyttää tällaiselta:



Korjaa virhe

Voit korjata virheitä askellustilassa, mutta jos tekemäsi muutokset ovat lauseissa, jotka on jo suoritettu, korjaukset eivät vaikuta ennen kuin lauseet suoritetaan seuraavan kerran. Testaa ehtorakenteeseen tekemäsi muutoksen vaikutus napsauttamalla Testaa-painiketta uudelleen.

18 Napsauta Askel-painiketta kolmesti.

Visual Basic suorittaa ehtorakenteen loppuun.

19 Napsauta Debug-työkalurivin Käynnistä-painiketta (tällä kertaa se toimii Jatka-painikkeena), jotta ohjelman suoritus jatkuu normaalinopeudella.

If-virhe-lomake tulee näkyviin.

20 Testaa tekemäsi korjaus napsauttamalla Testaa-painiketta. Ohjelma ilmoittaa sinun olevan teini-ikäinen, niin kuin pitikin. Lopeta ohjelma napsauttamalla Lopeta-painiketta.

21 Poista Debug-työkalurivi näkyvistä ja sulje Watches-ikkuna napsauttamalla niiden sulkemispainikkeita.

Onnittelut! Nyt osaat etsiä ja korjata ohjelman loogisia virheitä askellustilassa. Kun työskentelet Visual Basicin parissa, sinun kannattaa analysoida tekemiäsi ohjelmia käyttämällä askellustilaa ja Debug-työkaluriviä.

Lisätietoja: Askellustilan valitseminen Stop-lauseella

Voit siirtää Visual Basicin askellustilaan Stop-lauseella.

Jos tiedät tarkasti, missä ohjelman kohdassa haluat valita askellustilan ja aloittaa testaamisen, kirjoita kyseiseen kohtaan Stop-lause. Se keskeyttää ohjelman suorituksen ja avaa Code-ikkunan. Edellisessä esimerkissä olisit voinut valita askellustilan kirjoittamalla Stop-lauseen Command1_Click-aliohjelman alkuun tällä tavoin:

```
Private Sub Command1_Click()
    Stop    'valitse askellustila
    Age = Text1.Text

    If Age > 13 And Age < 20 Then
        Text2.Text = "Olet teini-iässä."
    Else
        Text2.Text = "Et ole teini-iässä."
    End If
End Sub
```

Kun käynnistät Stop-lauseen sisältävän ohjelman Visual Basic valitsee askellustilan, kun se kohtaa Stop-lauseen. Askellustilassa voit käyttää Code-ikkunaa ja Debug-työkaluriviä aivan samoin kuin keskeytettyäsi ohjelman komennolla. Kun olet testannut ohjelman, poista Stop-lause.

Jos haluat jatkaa seuraavaan harjoitukseen

➤ Jätä Visual Basic käyntiin ja siirry harjoitukseen 6.

Jos haluat lopettaa Visual Basicin

➤ Valitse File-valikosta komento Exit.

Jos näyttöön tulee Save-valintaikkuna, napsauta Yes.

Harjoituksen tiivistelmä

Tehtävä	Ohje	Painike
Muodostaa ehtolauseke	Kirjoita kaksi arvoa tai lauseketta ja niiden väliin vertailuoperaattori.	
Käyttää ehtorakennetta	Käytä If…Then- tai Select Case -rakennetta ja sen lausekkeita ja avainsanoja.	
Käyttää ehtolausekkeessa kahta vertailua	Lisää vertailujen väliin looginen operaattori (And, Or, Not tai Xor).	
Näyttää Debug-työkalurivi	Avaa View-valikon Toolbars-alivalikko ja valitse siitä komento Debug.	
Valita askellustila testausta varten	Napsauta Debug-työkalurivin Keskeytä-painiketta. *tai* Kirjoita ohjelmaan Stop-lause kohtaan, josta alkaen haluat käyttää askellustilaa.	
Suorittaa yksi lause Code-ikkunassa	Napsauta Askel-painiketta. *tai* Valitse Debug-valikosta komento Step Into.	
Tutkia muuttujaa Code-ikkunassa	Valitse tutkittava muuttuja ja napsauta Debug-työkalurivin Pikaseuranta-painiketta tai valitse Debug-valikosta komento Quick Watch.	
Poistaa seurantalauseke	Napsauta lauseketta Watches-ikkunassa ja napsauta Delete.	

Kun tarvitset tietoja aiheesta	Valitse Visual Basicin Help-valikosta Microsoft Visual Basic Help Topics, valitse Index-välilehti ja...
Vertailuoperaattorit	Kirjoita **comparison operators**.
Loogiset operaattorit	Kirjoita **logical operators**.
If…Then-ehtorakenteet	Kirjoita **If**.
Select Case -ehtorakenteet	Kirjoita **Select statement**.
Testaus	Kirjoita **debugging**.

Tietoja seuraavasta harjoituksesta

Seuraavassa harjoituksessa opit, miten ohjelmalla suoritetaan toistuvia tehtäviä. Näet myös, miten käytetään For...Next-, Do- ja While-lauseita ja Timer-objektia.

Harjoitus

6

Silmukoiden ja ajastimien käyttö

Arvioitu aika
50 min.

Tässä harjoituksessa opit

- toistamaan lauseet tietyn määrän kertoja For…Next-silmukalla
- näyttämään tietoja lomakkeessa Print-metodilla
- toistamaan lauseita Do-silmukassa, kunnes annettu ehto toteutuu
- suorittamaan lauseita tietyn ajan käyttämällä ajastinobjektia.

Harjoituksessa 5 näit, miten If...Then- ja Select Case -ehtorakenteilla valitaan suoritettavat ohjelmalauseet. Tässä harjoituksessa kuvaan, miten ohjelma suorittaa joukon lauseita yhä uudelleen *silmukassa*. Näytän, miten For...Next-silmukka toistaa lauselohkon tietyn määrän kertoja ja miten Do-silmukka toistaa lauselohkoa, kunnes annettu ehtolauseke toteutuu eli saa arvon True. Kerron myös, miten Print-metodilla näytetään tekstiä ja lukuja lomakkeessa ja miten ajastinobjektilla suoritetaan lausekkeita tiettyinä ajankohtina.

For…Next-silmukoiden kirjoittaminen

For...Next-silmukka toistaa lauselohkon niin monta kertaa kuin halutaan. Siitä on hyötyä, kun suoritetaan useita samantapaisia laskutoimituksia, muokataan näytön osia tai käsitellään käyttäjän antamia tietoja. For...Next-silmukka on oikeastaan vain lyhyt tapa kirjoittaa pitkä luettelo ohjelmalauseita. Jos luettelossa on samat lauseet moneen kertaan, voit yksinkertaisesti kirjoittaa ne vain kerran For...Next-silmukan sisäpuolelle ja määrätä, montako kertaa ne toistetaan.

For...Next-silmukka muodostetaan näin:

```
For muuttuja = alku To loppu
    toistettavat lauseet
Next muuttuja
```

For...Next-rakenteessa alku- ja loppuarvot määräävät, montako kertaa silmukka toistetaan.

Tässä For, To ja Next ovat välttämättömiä avainsanoja, ja yhtäläisyysoperaattori (=) on pakollinen. Korvaa *muuttuja* sen muuttujan nimellä, johon silmukkalaskuri tallennetaan, ja *alku* ja *loppu* lukuarvoilla, jotka määrittävät muuttujan alku- ja loppuarvon. For- ja Next-lauseiden väliset rivit suoritetaan joka kerta, kun silmukka toistetaan.

Esimerkiksi seuraava For...Next-silmukka antaa tietokoneen kaiuttimella neljä äänimerkkiä nopeasti peräkkäin:

```
For i = 1 To 4
    Beep
Next i
```

Tämä silmukka vastaa Beep-lauseen kirjoittamista neljään kertaan. Visual Basic -kääntäjän kannalta se on sama kuin lauseet

```
Beep
Beep
Beep
Beep
```

Silmukassa käytetään muuttujaa i, joka on perinteisesti For...Next-silmukan kokonaislukulaskurin nimi. Silmukan joka kierroksella laskurimuuttujan arvo kasvaa yhdellä. (Ensimmäisellä kierroksella muuttujan arvo on 1 eli *alku*, ja viimeisellä kierroksella 4 eli *loppu*.) Kuten seuraavista esimerkeistä näet, laskurimuuttujasta on paljon hyötyä silmukassa.

Laskurimuuttujan näyttäminen Print-metodilla

Laskurimuuttuja on aivan kuin mikä tahansa muuttuja. Se voidaan sijoittaa ominaisuuteen, sitä voidaan käyttää laskutoimituksessa ja sen arvo voidaan tuoda näkyviin. Yksi kätevimpiä tapoja näyttää laskurimuuttujan arvo on käyttää Print-metodia, erikoislausetta, joka esittää tietoja lomakkeessa tai lähettää ne tulostimeen. Print-metodia kutsutaan tällaisella lauseella:

```
Print lauseke
```

Print-metodi lähettää tietoja lomakkeeseen tai tulostimeen.

Lauseke on muuttuja, ominaisuus, merkkijono tai lukuarvo. Seuraavassa ohjelmassa sinun pitää käyttää Print-metodia, kun näytät For...Next-silmukan tulokset lomakkeessa.

VIHJE Jos aiot välillä pienentää Print-metodilla näytetyt tiedot sisältävän lomakkeen kuvakkeeksi, määritä lomakkeen AutoRedraw-ominaisuudeksi True. Tällöin Visual Basic muodostaa näytettävät tiedot uudelleen, kun palautat lomakkeen ennalleen. Lomakkeen muut objektit piirtyvät automaattisesti uudelleen, mutta Print-metodilla näytetty teksti tulee näkyviin vain, jos AutoRedraw-ominaisuus on True.

Tietojen näyttäminen For...Next-silmukalla

Komentopainike-työkalu

1. Käynnistä Visual Basic.

   Jos Visual Basic on käynnissä, luo EXE-projekti.

2. Suurenna lomaketta hiirellä, jotta siinä on tarpeeksi tilaa tietojen esittämistä varten.

3. Valitse Komentopainike-työkalu ja piirrä lomakkeen oikeaan reunaan komentopainike.

4. Avaa Properties-ikkuna ja määritä komentopainikkeen Caption-ominaisuudeksi ”Silmukka”.

5. Avaa Properties-ikkunan yläreunassa oleva objektiluettelo ja valitse objekti Form1.

   Lomakkeen ominaisuudet näkyvät Properties-ikkunassa.

6. Vaihda Font-ominaisuudeksi Times New Roman.

   Font-ominaisuus määrää, millaisena teksti näkyy lomakkeessa. Voit käyttää mitä tahansa järjestelmään asennettua fonttia, mutta TrueType-fontit toimivat parhaiten, koska niiden kokoa voidaan muuttaa ja koska ne ovat näytössä samanlaiset kuin paperilla.

7. Vaihda AutoRedraw-ominaisuudeksi True.

   Jos välillä piilotat lomakkeen, Visual Basic kirjoittaa uudelleen tekstin, joka on näytetty Print-metodilla.

8. Kaksoisnapsauta lomakkeen Silmukka-painiketta.

   Tapahtuma-aliohjelma Command1_Click ilmestyy Code-ikkunaan.

9. Kirjoita aliohjelmaan tällaiset lauseet:

```
For i = 1 To 10
    Print "Rivi"; i
Next i
```

   Tämä silmukka näyttää Print-metodilla sanan *Rivi* ja silmukkalaskurin arvon kymmenen kertaa. Print-lauseen puolipiste (;) määrää, että muuttujan arvon ja Line-sanan väliin edelle ei jätetä tyhjää. (Voit kuitenkin havaita, että ohjelma lisää luvun eteen välin. Kun Print-metodi näyttää lukuarvoja, se varaa tilaa miinusmerkille, vaikka tätä ei olisikaan.)

Valmis ForLoop-ohjelma on kansiossa \Vb5Sbs\Less06.

HUOMAA Print-metodin lausekeluettelon osat voidaan erotella puolipistein (;) tai pilkuin (,). Puolipiste sijoittaa osat peräkkäin, kun taas pilkku jättää niiden väliin yhden sarakkeen. Voit käyttää lausekeluettelossa millaista tahansa puolipisteiden ja pilkkujen yhdistelmää.

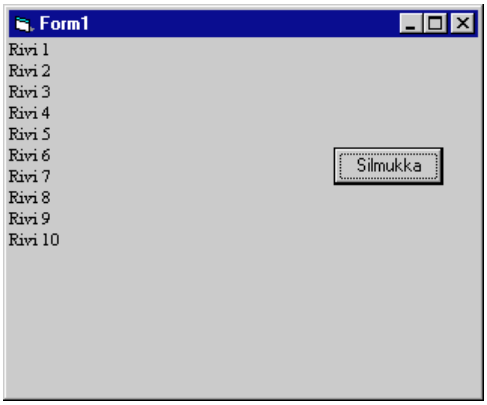
Nyt voit käynnistää ohjelman.

10 Napsauta työkalurivin Käynnistä-painiketta.

Käynnistä-painike

11 Napsauta Loop-painiketta.

For…Next-silmukka näyttää lomakkeesssa 10 tekstiriviä näin:



12 Napsauta Silmukka-painiketta uudelleen.

For…Next-silmukka lisää lomakkeeseen toiset 10 riviä (tai niin monta kuin mahtuu). Aina kun yksi rivi on kirjoitettu, kohdistin siirtyy seuraavalle riville, kunnes se saavuttaa lomakkeen alareunan.

Lopeta-painike

13 Lopeta ohjelma napsauttamalla työkalurivin Lopeta-painiketta.

Tallenna projekti -painike

14 Napsauta työkalurivin Tallenna projekti -painiketta. Tallenna lomake nimellä **MyForLoop.frm** ja projekti nimellä **MyForLoop.vbp**.

Sijoita tiedostot kansioon \Vb5Sbs\Less06.

Ominaisuuden muuttaminen For…Next-silmukassa

Voit muuttaa lomakkeen fonttien pistekokoa FontSize-ominaisuudella.

Ominaisuuksien ja muuttujien arvoja on mahdollista muuttaa silmukassa. Seuraavassa esimerkissä muokkaat MyForLoop-ohjelmaa niin, että se muuttaa FontSize-ominaisuutta For...Next-silmukassa. FontSize-ominaisuus määrää lomakkeen tekstin pistekoon; se on vaihtoehto pistekoon muuttamiselle Font-ominaisuudella.

FontSize-ominaisuuden muuttaminen

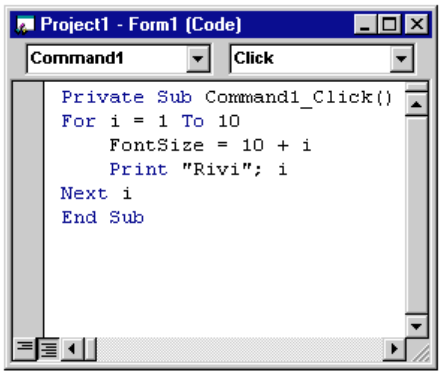
1. Avaa tapahtuma-aliohjelma Command1_Click.

   For…Next-silmukka näkyy Code-ikkunassa.

2. Lisää heti For-lauseen jälkeen tällainen lause:

```
FontSize = 10 + i
```

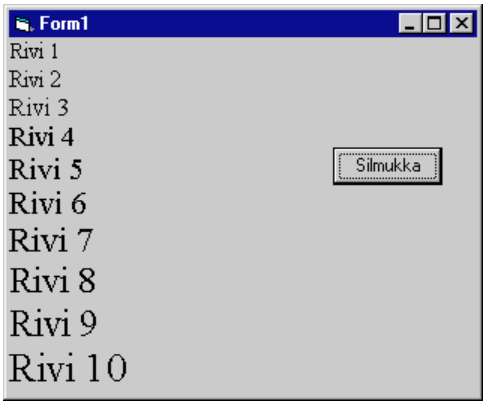
   Lause määrittää lomakkeen FontSize-ominaisuuden kymmentä suuremmaksi kuin silmukkalaskurin arvo. Silmukan ensimmäisellä kierroksella fonttikoko on siis 11 pistettä, toisella kierroksella 12 pistettä ja viimeisellä kierroksella 20 pistettä. Valmiin For...Next-silmukan pitäisi näyttää tällaiselta:



Käynnistä-painike

3. Käynnistä ohjelma napsauttamalla työkalurivin Käynnistä-painiketta.
4. Napsauta Silmukka-painiketta.

   For…Next-silmukka näyttää lomakkeessa tällaiset tekstit:



Lomakkeen fonttikoko kasvaa silmukan joka kierroksella.

Lopeta-painike

5 Lopeta ohjelma napsauttamalla Lopeta-painiketta.

Ohjelman suoritus päättyy, ja ohjelmointiympäristö palaa näkyviin.

6 Valitse File-valikosta komento Save MyForLoop.frm As. Tallenna muokattu lomake nimellä **MyGrowFont.frm**.

Valmis Grow-Font-ohjelma on kansiossa \Vb5Sbs\Less06.

7 Valitse File-valikosta komento Save Project As. Tallenna projekti nimellä **MyGrowFont.vbp**.

TÄRKEÄÄ For…Next-silmukalla voidaan säästää paljon tilaa ohjelmassa. Edellisen esimerkin 4 lauseen silmukka vastaa 20 ohjelmalausetta.

Monimutkaisemmat For…Next-silmukat

Voit muuttaa For...Next-silmukka-muuttujan peräkkäisten arvojen väliä avainsanalla Step.

For...Next-silmukan silmukkalaskurille on paljon käyttöä ohjelmissa. Voit helposti muodostaa erilaisia lukusarjoja. Jos tarvitset muunlaista lukusarjaa kuin 1, 2, 3 ja niin edelleen, määritä alku- ja loppuarvot sekä silmukkalaskurin muutosaskel Step-avainsanalla. Esimerkiksi silmukka

```
For i = 5 To 25 Step 5
    Print i
Next i
```

näyttäisi seuraavan lukusarjan:

```
5
10
15
20
25
```

Voit käyttää muutosaskelena myös desimaalilukua. Esimerkiksi silmukka

Step-avainsanassa saa olla myös desimaaliarvoja.

```
For i = 1 To 2.5 Step 0.5
    Print i
Next i
```

näyttäisi seuraavan lukusarjan:

```
1
1.5
2
2.5
```

Silmukalla voidaan paitsi näyttää silmukkalaskurin arvo, myös asettaa ominaisuuksia, suorittaa laskutoimituksia ja käsitellä tiedostoja. Seuraavassa ohjelmassa käytät silmukkamuuttujaa, kun avaat Visual Basicin kuvaketiedostoja, joiden nimissä esiintyy numeroita. Ohjelma havainnollistaa myös sitä, miten For...Next-silmukalla käsitellään useita kuvaobjekteja ryhmänä. Jotta voisit muokata kuvaobjekteja tehokkaasti, muodosta niistä objektitaulukko.

Tiedostojen avaaminen For...Next-silmukalla

1. Valitse File-valikosta komento New Project ja napsauta OK.
2. Napsauta Kuva-työkalua ja piirrä lomakkeen vasempaan yläkulmaan pieni neliömäinen kuvaobjekti.

Kuva-työkalu

3. Valitse Edit-valikosta komento Copy.

   Kuvaobjektin kopio tallentuu Windowsin Leikepöydälle. Lisää sen avulla lomakkeeseen kolme muuta kuvaobjektia.

4. Valitse Edit-valikosta komento Paste.

   Objektitaulukko muodostetaan kopioimalla ja liittämällä objekteja.

   Visual Basic kysyy, haluatko luoda ohjelmaan objektitaulukon. *Objektitaulukko* on käyttöliittymän samanlaisten osien kokoelma. Taulukon jokaisella objektilla on sama nimi, joten taulukko voidaan valita ja määrittää kerralla. Toisaalta voit viitata taulukon objekteihin myös erikseen, joten käyttöliittymän kaikki osat ovat täydellisesti hallittavissa.

5. Muodosta objektitaulukko napsauttamalla Yes.

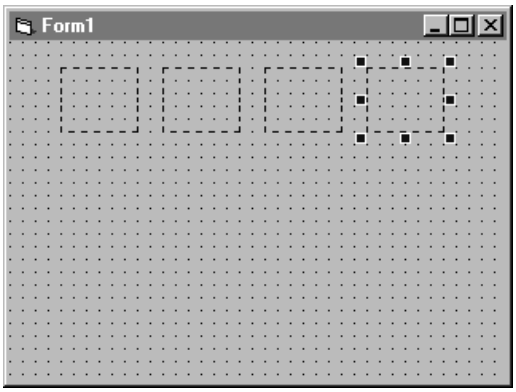
   Visual Basic luo kuvaobjektien taulukon ja liittää piirtämäsi kuvaobjektin kopion lomakkeen vasempaan yläkulmaan. Uusi kuvaobjekti on valmiiksi valittu.

6. Vedä toinen kuvaobjekti ensimmäisen oikealle puolelle.

HUOMAA Kun olet valinnut objektin, voit vetää sen mihin tahansa kohtaan lomakkeessa.

7. Valitse Edit-valikosta komento Paste ja vedä kolmas kuvaobjekti toisen oikealle puolelle.
8. Valitse vielä kerran Paste-komento ja siirrä neljäs kuvaobjekti kolmannen oikealle puolelle.

   Kun olet liittänyt kaikki kopiot lomakkeeseen, sen pitäisi olla seuraavan sivun kuvan mukainen.



Komentopainike-työkalu

Voit käsitellä objektitaulukon kaikkia objekteja valitsemalla ne ryhmänä.

9 Napsauta Komentopainike-työkalua ja piirrä lomakkeen alaosaan komentopainike.

Määritä seuraavaksi lomakkeen objektien ominaisuudet. Muokkaa ensin kuvaobjekteja ryhmänä.

10 Napsauta ensimmäistä kuvaobjektia, pidä VAIHTONÄPPÄINTÄ alhaalla ja napsauta toista, kolmatta ja neljättä kuvaobjektia.

Objektiryhmän kuvaobjektit on nyt valittu.

11 Avaa Properties-ikkuna ja määritä seuraavat ominaisuudet. (Kun olet määrittänyt kuvakeobjektien ominaisuudet, napsauta komentopainikeobjektia.)

Objekti	Ominaisuus	Asetus
Image1-objektitaulukko	BorderStyle	1 - Fixed Single
	Stretch	True
Command1	Caption	”Näytä kuvakkeet”

12 Kaksoisnapsauta lomakkeen Näytä kuvakkeet -painiketta, jotta näet komentopainikeobjektin tapahtuma-aliohjelman.

Tapahtuma-aliohjelma Command1_Click ilmestyy Code-ikkunaan.

13 Levennä Code-ikkunaa ja kirjoita siihen tällainen silmukka:

```
For i = 1 To 4
    Image1(i - 1).Picture = _
      LoadPicture("c:\vb5sbs\less06\misc0" & i & ".ico")
Next i
```

HUOMAA Tämä LoadPicture-funktion kutsu on liian pitkä kirjoitettavaksi tämän kirjan yhdelle riville, joten jaoin sen kahtia käyttäen rivinjatkomerkkiä (_). Voit kirjoittaa tämän merkin mihin tahansa kohtaan lausekkeessa paitsi merkkijonolausekkeen sisään.

Silmukka lataa lomakkeeseen LoadPicture-funktiolla neljä kuvaketiedostoa kiintolevyn kansiosta \Vb5Sbs\Less06. Silmukan vaikea osa on lause

```
Image1(i - 1).Picture = _
  LoadPicture("c:\vb5sbs\less06\misc0" & i & ".ico")
```

joka lukee kuvaketiedostot levyltä. Lauseen ensimmäinen osa eli

```
Image1(i - 1).Picture
```

käsittelee vuorollaan objektitaulukon kunkin kuvaobjektin Picture-ominaisuuden. Objektitaulukon alkioihin viitataan indeksinumeroilla, joten tässä esimerkissä erilliset kuvaobjektit ovat Image1(0), Image1(1), Image1(2) ja Image1(3). Sulkein ympäröity numero on objektin indeksinumero taulukossa. Esimerkissä oikea indeksinumero saadaan vähentämällä silmukkamuuttujan arvosta 1.

Kuvaketiedoston nimi muodosteaan käyttämällä silmukkamuuttujaa ja harjoituksessa 4 kuvattua yhteenliittämisoperaattoria. Lauseke

```
LoadPicture("c:\vb5sbs\less06\misc0" & i & ".ico")
```

yhdistää polkunimen, tiedostonimen ja tarkenteen .ico, joista muodostuu neljä sallittua tiedostonimeä. Kiintolevyllä on valmiina tiedostot Misc01.ico, Misc02.ico, Misc03.ico ja Misc04.ico, jotka tämä aliohjelma lataa muistiin. Aliohjelma toimii, koska kansiossa \Vb5Sbs\Less06 on useita tiedostoja, joiden nimet noudattavat mallia Misc*xx*.ico. Kun huomaat tämän nimimallin, osaat muodostaa tiedostot lukevan For...Next-silmukan.

Tallenna projekti –painike

14 Napsauta työkalurivin Tallenna projekti -painiketta.

Tallenna lomake nimellä **MyCtlArray.frm** ja projekti nimellä **MyCtlArray.vbp.**

Käynnistä–painike

15 Käynnistä ohjelma napsauttamalla työkalurivin Käynnistä-painiketta ja napsauta sitten Näytä kuvakkeet -painiketta.

Valmis CtlArray–ohjelma on kansiossa \Vb5Sbs\Less06.

For…Next-silmukka lukee kuvaketiedostot kiintolevyltä ja näyttää niiden sisällöt lomakkeen kuvaobjekteissa.

HUOMAA Jos Visual Basic näyttää virheilmoituksen, tarkista, ettei ohjelmassa ole kirjoitusvirheitä ja että kuvaketiedostot ovat määrittämässäsi hakemistossa. Jos asensit tämän kirjan harjoitustiedostot johonkin muuhun kuin oletuskansioon tai olet siirtänyt kuvaketiedostot muualle asentamisen jälkeen, tapahtuma-aliohjelman polkunimi ei ehkä ole oikea.

Ohjelman tulokset ovat tällaiset:

16 Lopeta ohjelma napsauttamalla sen otsikkorivin sulkemispainiketta.

Ohjelman suoritus päättyy, ja ohjelmointiympäristö palaa näkyviin.

Step-avainsanan käyttö MyCtlArray-ohjelmassa

Kuvittele, että kansio \Vb5Sbs\Less06 on täynnä tiedostoja, joiden nimet noudattavat mallia Misc*xx*.ico, ja haluat näyttää kuvaobjekteissa hieman enemmän kuvakkeita. Kun tarkoitus on ladata tiedostot, joiden nimet ovat eri mallin mukaiset, sinun pitää muuttaa For-lauseessa olevia lukuja sekä lausetta, joka muodostaa objektitaulukon indeksin ja kuvaketiedoston nimen. Sinun pitää myös muuttaa kuvaobjektien indeksit silmukkalaskurin uusien arvojen mukaisiksi.

MyCtlArray-ohjelman muokkaaminen

1 Napsauta lomakkeen ensimmäistä kuvakeobjektia ja avaa Properties-ikkuna.

Kuvaobjektin Image1(0) ominaisuudet näkyvät Properties-ikkunassa.

2 Vaihda Index-ominaisuudeksi 22.

Objektitaulukon ensimmäisen alkion indeksi on yleensä 0, mutta voit vaihtaa sen, jos tämä helpottaa taulukon käsittelemistä. Tässä esimerkissä avattavien tiedostojen nimet ovat Misc22.ico, Misc24.ico, Misc26.ico ja and Misc28.ico. Kun vaihdat objektien indeksinumeroiksi 22, 24, 26 ja 28, objekteihin on helpompi viitata.

3 AvaaProperties-ikunan objektiluettelo ja valitse objekti Image1(1).

4 Vaihda toisen kuvaobjektin Index-ominaisuudeksi 24.

5 Avaa objektiluettelo ja valitse objekti Image1(2).

6 Vaihda kolmannen kuvaobjektin Index-ominaisuudeksi 26.

7 Avaa objektiluettelo ja valitse objekti Image1(3).

8 Vaihda neljännen kuvaobjektin Index-ominaisuudeksi 28.

Muokkaa seuraavaksi For…Next-silmukkaa.

9 Kaksoisnapsauta lomakkeen Näytä kuvakkeet -painiketta.

Tapahtuma-aliohjelma Command1_Click ilmestyy Code-ikkunaan.

10 Muuta For-lause tällaiseksi:

```
For i = 22 To 28 Step 2
```

Tämä lause sijoittaa silmukkamuuttujaan ensimmäisellä kierroksella arvon 22, toisella kierroksella arvon 24 ja sitten arvot 26 ja 28. Muokkaa vielä silmukassa olevaa LoadPicture-funktion kutsua.

11 Vaihda objektitaulukon indeksinumeron Image1(i – 1) tilalle Image1(i).

Koska indeksinumero vastaa nyt täsmälleen silmukkamuuttujan arvoja ja kuvaobjektien nimiä, sitä ei tarvitse johtaa millään laskutoimituksella.

12 Poista toinen 0 (nolla) LoadPicture-funktion kutsussa olevasta polkunimestä.

Misc*xx*.ico-tiedostojen nimissä ei ole nollia. Valmiin tapahtuma-aliohjelman pitäisi näyttää tällaiselta:

13 Napsauta työkalurivin Käynnistä-painiketta ja sitten lomakkeen Näytä kuvakkeet -painiketta.

For…Next-silmukka lataa uudet kuvakkeet tekstiobjekteihin. Lomakkeen pitäisi olla seuraavan sivun kuvan mukainen.

HUOMAA Kansio \Program Files\DevStudio\VB\Icons\Misc sisältää kymmeniä kiinnostavia kuvakkeita, jotka muistuttavat edellä esitettyjä. Näitä tiedostoja ei asenneta kiintolevylle automaattisesti. Jos kansio puuttuu, käynnistä Visual Basic 5:n asennusohjelma, valitse Add/Remove ja valitse vaihtoehto Icon Library. Voit vapaasti käyttää näitä kuvakkeita painikkeina tai koristeina ohjelmissasi.

14 Napsauta ohjelman otsikkorivillä olevaa sulkemispainiketta.

Ohjelman suoritus päättyy, ja ohjelmointiympäristö palaa näkyviin.

15 Tallenna muokattu lomake ja projekti nimellä **MyStepLoop**.

Valmis Step-Loop-ohjelma on kansiossa \Vb5Sbs\Less06.

Exit For -lauseet

Exit For -lauseella poistutaan For...Next-silmukasta ennen kuin se on suoritettu loppuun. Näin ohjelma voi reagoida tapahtumaan, joka sattuu ennen silmukan päättymistä. Esimerkki:

```
For i = 1 To 10
    InpName = _
       InputBox("Anna nimi tai lopeta kirjoittamalla Valmis.")
    If InpName = "Valmis" Then Exit For
    Print InpName
Next i
```

Tämä silmukka kysyy käyttäjältä kymmenen nimeä ja lisää ne lomakkeeseen. Jos käyttäjä kuitenkin kirjoittaa nimen sijasta tekstin *Valmis*, silmukka keskeytyy ja ohjelman suoritus jatkuu Next-lausetta seuraavasta lauseesta. Exit For -lausetta käytetään tavallisesti If-lauseen kanssa. Sillä voidaan hoitaa erikoistapaukset, jotka tulevat esiin silmukassa, esimerkiksi keskeyttäminen, kun tietty raja-arvo on saavutettu.

Do-silmukoiden kirjoittaminen

Do–silmukka toistaa lauseita, kunnes annettu ehto toteutuu.

Vaihtoehto For...Next-silmukalle on Do-silmukka, joka toistaa lauselohkoa, kunnes annettu ehto toteutuu. Do-silmukka on tärkeä tilanteessa, jossa toistojen määrää ei tiedetä etukäteen. Oletetaan, että ohjelman pitää kysyä käyttäjältä tietokantaan lisättäviä nimiä, kunnes käyttäjä kirjoittaa tekstiruutuun nimen sijasta sanan *Valmis*. Tässä tapauksessa ohjelman pitää pyöriä Do-silmukassa miten kauan tahansa, kunnes käyttäjä lopettaa kirjoittamalla merkkijonon ”Valmis .

Do-silmukalla on useita muotoja sen mukaan, missä ja miten silmukkaehto lasketaan. Yleisin muoto on tällainen:

```
Do While ehto
    suoritettava lauselohko
Loop
```

Esimerkiksi seuraava Do-silmukka lukee syöttötietoja, kunnes annetaan vastaukseksi sana *Valmis*:

```
Do While InpName <> "Valmis"
   InpName = InputBox("Anna nimi tai lopeta kirjoittamalla Valmis.")
   If InpName <> "Valmis" Then Print InpName
Loop
```

Ehtolausekkeen sijainti vaikuttaa Do–silmukan toimintaan.

Tämän silmukan ehtolause on InpName <> ”Valmis", jonka Visual Basic -kääntäjä tulkitsee näin: ”toista silmukkaa niin kauan kuin InpName-muuttujan arvo ei ole merkkijono *Valmis*”. Tässä kannattaa huomata kiinnostava seikka: jos Do-silmukan alussa oleva ehto ei ole tosi, kun silmukkaan saavutaan ensimmäisen kerran, silmukka jää kokonaan väliin. Jos siis InpName-muuttujassa olisi merkkijono ”Valmis” jo ennen silmukan alkamista (ehkäpä jäänteenä aliohjelman aiemmasta sijoituslauseesta), Visual Basic hyppäisi koko silmukan ohi ja jatkaisi Loop-avainsanaa seuraavasta lauseesta. Huomaa myös, että tällaisessa silmukassa tarvitaan ylimääräinen If..Then-ehto, joka estää silmukan lopettavan tiedon näyttämisen.

Jos silmukka on suoritettava vähintään kerran, sijoita ehtolause sen loppuun. Esimerkiksi silmukka

```
Do
   InpName = InputBox("Anna nimi tai lopeta kirjoittamalla Valmis.")
   If InpName <> "Valmis" Then Print InpName
Loop While InpName <> "Valmis"
```

on periaatteessa sama kuin edellinen, mutta ehto tutkitaan vasta, kun nimi on saatu InputBox-funktiosta. Näin InpName-muuttujan arvo päivittyy ennen silmukkaehdon testaamista, ja muuttujaan aiemmin tallennettu merkkijono ”Valmis” ei aiheuta silmukan ohittamista. Kun ehto tutkitaan viimeiseksi, silmukka suoritetaan aina vähintään kerran, mutta sinun pitää usein lisätä muutama ylimääräinen lause tietojen käsittelemistä varten.

Loputtoman silmukan estäminen

Varmista, että jokaisessa silmukassa on oikea lopetusehto.

Koska tietokone voi toistaa Do-silmukkaa miten pitkään tahansa, on erittäin tärkeää määrittää lopetusehto niin, että silmukka varmasti päättyy. Jos ehtolauseke ei koskaan saa arvoa False, silmukka toistuu loputtomiin, ja ohjelma ei vastaa näppäilyihin eikä hiiren liikkeisiin. Katso tätä esimerkkiä:

```
Do
    Number = InputBox("Anna neliöitävä luku. Luku -1 lopettaa.")
    Number = Number * Number
    Print Number
Loop While Number >= 0
```

Tässä silmukassa käyttäjä kirjoittaa lukuja toisensa perään, ja ohjelma korottaa ne toiseen potenssiin ja näyttää ne lomakkeessa. Käyttäjä ei kuitenkaan pysty lopettamaan ohjelmaa, koska lopetusehto ei toimi. Kun käyttäjä antaa luvun –1, ohjelma korottaa sen toiseen ja sijoittaa tuloksen eli luvun 1 Number-muuttujaan. (Virhe voidaan poistaa muuttamalla lopetusehtoa.) Loputtomia Do-silmukoita on syytä varoa. Ne on onneksi melko helppo löytää, jos testaat ohjelmasi huolellisesti.

Seuraavassa ohjelmassa näytetään, miten muunnetaan fahrenheitasteita celsiusasteiksi Do-silmukassa. Ohjelma kysyy käyttäjältä lämpötilan InputBox-funktiolla, muuntaa sen toiseen yksikköön ja näyttää tuloksen ilmoitusikkunassa. Ohjelma kuvaa myös sitä, miten lomake muutetaan näkymättömäksi määrittämällä sen Visible-ominaisuudeksi False.

Lämpötilojen muuntaminen Do-silmukassa

1. Valitse File-valikosta komento New Project ja napsauta OK.

   Visual Basic näyttää uuden lomakkeen ohjelmointiympäristössä.

2. Avaa Properties-ikkuna ja määritä lomakkeen Visible-ominaisuudeksi False.

   Voit piilottaa lomakkeen ohjelman suorituksen aikana määrittämällä sen Visible-ominaisuudeksi False.

   Kun asetat lomakkeen Visible-ominaisuudeksi False, Visual Basic piilottaa lomakkeen käynnistettyäsi ohjelman. Näin koko käyttöliittymä on näkymätön suorituksen aikana — mitään objektia ei voi näyttää. Et varmaankaan tarvitse tätä mahdollisuutta usein, mutta siitä on hyötyä, jos ohjelman tai sen osan pitää suorittaa jotakin tehtävää taustalla. Koska tämä ohjelma vain lukee fahrenheitasteita ja näyttää celsiusasteita, sen lomakkeen piilottaminen on hyvä ajatus. Voit lukea tiedot InputBox-funktiolla ja näyttää ne MsgBox-funktiolla.

3. Kaksoisnapsauta lomaketta.

   Tapahtuma-aliohjelma Form_Load suoritetaan, kun ohjelma käynnistyy.

   Tapahtuma-aliohjelma Form_Load ilmestyy Code-ikkunaan. Siihen sijoitetaan kaikki tämän ohjelman lauseet.

4. Kirjoita seuraavat ohjelmalauseet:

```
Prompt = "Anna  lämpötila  fahrenheitasteina."
Do
    FTemp = InputBox(Prompt, _
                  "Fahrenheitasteista  celsiusasteiksi")
    If FTemp <> "" Then
        Celsius = Int((FTemp + 40) * 5 / 9 - 40)
        MsgBox (Celsius), , "Lämpötila  celsiusasteina"
    End If
Loop While FTemp <> ""
End
```

Nämä yhdeksän lausetta hoitavat ohjelman laskutoimitukset. Ensimmäinen lause sijoittaa merkkijonon Prompt-muuttujaan, joka sitten näytetään ohjeeksi valintaikkunassa. Do-silmukka kysyy yhä uudelleen fahrenheitasteina annettuja lämpötiloja, muuntaa ne celsiusasteiksi ja näyttää tulokset MsgBox-funktiolla. Silmukka toistuu, kunnes käyttäjä napsauttaa Cancel-painiketta, mikä aiheuttaa tyhjän merkkijonon sijoittamisen Ftemp-muuttujaan. Silmukka tarkistaa tyhjän arvon While-lauseella. Ohjelmalause

```
Celsius = Int((FTemp + 40) * 5 / 9 - 40)
```

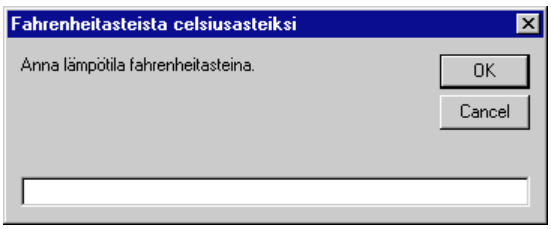
tekee varsinaisen muunnoksen fahrenheitasteista celsiusasteiksi. Se perustuu tavanomaiseen muunnoskaavaan, mutta se poistaa tuloksesta desimaaliosan Int-funktiolla. Tämä heikentää tuloksen tarkkuutta, mutta toisaalta estää turhan pitkien lukujen näyttämisen. Muuten esimerkiksi 70 fahrenheitastetta olisi 21.1111111111111 celsiusastetta.

Kokeile ohjelmaa.

5 Napsauta työkalurivin Käynnistä-painiketta.

Käynnistä-painike

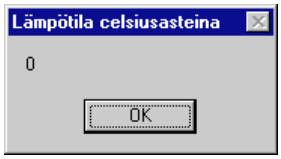
Ohjelma käynnistyy, ja InputBox-funktio kysyy fahrenheitasteina annetun lämpötilan (lomake on näkymätön). Valintaikkunan pitäisi näyttää tällaiselta:



6 Kirjoita 32 ja napsauta OK.

32 fahrenheitastetta on 0 celsiusastetta, kuten seuraavasta ilmoitusikkunasta näkyy:

Valmis Celsius-ohjelma on kansiossa \Vb5Sbs\Less06.



7 Napsauta OK. Kirjoita tekstiruutuun 72 ja napsauta OK.

72 fahrenheitastetta on 22 celsiusastetta.

8 Napsauta OK ja lopeta ohjelma napsauttamalla Cancel.

Ohjelman suoritus päättyy, ja ohjelmointiympäristö palaa näkyviin.

Tallenna projekti -painike

9 Napsauta Tallenna projekti -painiketta ja tallenna lomake ja projekti nimellä **MyCelsius**.

Until-avainsanan käyttö Do-silmukoissa

Tähän mennessä käyttämäsi Do-silmukat ovat toistaneet lauseita *niin kauan kuin* silmukkaehto on ollut tosi. Do-silmukassa voi olla While-avainsanan sijasta Until-avainsana, joka määrää, että silmukka toistuu, *kunnes* silmukkaehto toteutuu. Until-avainsanakin voidaan sijoittaa silmukan alkuun tai loppuun. Seuraava Do-silmukka perustuu Until-avainsanaan, joten silmukka toistuu, kunnes käyttäjä kirjoittaa *Valmis*:

```
Do
    InpName = _
      InputBox("Anna nimi tai lopeta kirjoittamalla Valmis.")
    If InpName <> "Valmis" Then Print InpName
Loop Until InpName = "Valmis"
```

Until-avainsanaa käyttävä silmukka on siis hyvin samanlainen kuin While-avainsanaa käyttävä, paitsi että ehtolausekkeessa on päinvastainen operaattori — tässä esimerkissä yhtä suuri kuin (=) eikä erisuuri kuin (<>). Jos Until-avainsana vaikuttaa sinusta hyödylliseltä, sovella sitä Do-silmukoiden ehtoihin.

Ajastinobjektien käyttö

Ajastinobjekti toimii kuin ohjelman näkymätön sekuntikello.

Visual Basic -ohjelma voi suorittaa lauselohkoa *tietyn ajan* käyttämällä ajastinobjektia. Ajastinobjekti on näkymätön sekuntikello, jolla voit hyödyntää järjestelmäkelloa ohjelmissasi. Se toimii kuin munakello eli laskee jäljellä olevaa aikaa, aiheuttaa viivästyksen ohjelmassa tai toistaa toiminnon määritetyin välein.

Ajastinobjektin tarkkuus on millisekunti eli sekunnin tuhannesosa. Vaikka ajastinobjektit eivät näy ohjelmaa suoritettaessa, kullakin ajastimella on tapahtuma-aliohjelma, joka suoritetaan aina määritetyn ajan kuluttua. Aika asetetaan Interval-ominaisuudella, ja ajastin otetaan käyttöön sijoittamalla sen Enabled-ominaisuudeksi True. Kun ajastin on käytössä, se toimii koko ajan eli suorittaa tapahtuma-aliohjelmansa annetuin välein, kunnes käyttäjä pysäyttää ohjelman tai kunnes ajastin poistetaan käytöstä.

Digitaalikellon ohjelmoiminen käyttämällä ajastinobjektia

Ajastinobjektilla voidaan tehdä vaikkapa digitaalikello. Seuraava tehtäväsi on kirjoittaa yksinkertainen digitaalikello-ohjelma, joka näyttää ajan sekunnin tarkkuudella. Määritä ajastimen Interval-ominaisuudeksi 1000, jotta Visual Basic päivittää näytetyn ajan joka 1000. millisekunti eli kerran sekunnissa. Koska Windows on moniajokäyttöjärjestelmä ja muutkin ohjelmat voivat vaatia siltä käsittelyaikaa, Visual Basic ei ehkä pysty päivittämään kelloa joka sekunti, mutta se ottaa mahdollisen eron aina kiinni. Jos haluat ajan muuttuvan useammin, esimerkiksi joka kymmenes sekunti, muuta Interval-ominaisuuden arvoa.

Interval-ominaisuus määrittää ajastimen nopeuden.

DigClock-ohjelman luominen

1. Valitse File-valikosta komento New Project ja napsauta OK.
2. Pienennä lomaketta hieman.

   Kello ei tarvitse paljon tilaa näytöstä.
3. Napsauta Ajastin-työkalua.

Ajastin-työkalu

4. Lisää lomakkeen vasempaan reunaan pieni ajastinobjekti.

   Visual Basic muuttaa objektin vakiokokoiseksi.

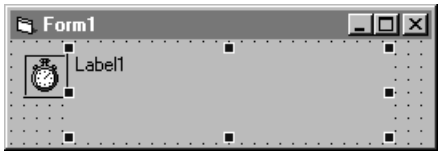
Ajastinobjektin koko on aina sama.

5. Napsauta Teksti-työkalua.

Teksti-työkalu

6. Lisää lomakkeen keskelle tekstiruutu, joka varaa suurimman osan lomakkeen alasta.

   Tähän tekstiruutuun sijoitetaan kellonaika. Lomakkeen pitäisi näyttää tällaiselta:



Lomakkeen Caption-ominaisuus määrää ohjelman otsikkorivin tekstin.

7 Avaa Properties-ikkuna ja määritä objektien ominaisuudet taulukon mukaisesti. Anna DigClock-ohjelmalle nimi, joka näkyy sen otsikkorivillä, asettamalla Form1-objektin Caption-ominaisuudeksi "Digitaalikello".

Objekti	Ominaisuus	Asetus
Label1	Caption	(Tyhjä)
	Font	Times New Roman, Bold, 24 pistettä
	Alignment	2 - Center
Timer1	Interval	1000
	Enabled	True
Form1	Caption	"Digitaalikello"

HUOMAA Jos haluat lisätä kellon taustalle kuvan, sijoita Form1-objektin Picture-ominaisuuteen grafiikkatiedoston polkunimi.

Valmis DigClock-ohjelma on kansiossa \Vb5Sbs\Less06.

Kirjoita ajastinobjektiin kuuluvat lauseet.

8 Napsauta lomakkeen ajastinobjektia.

Tapahtuma-aliohjelma Timer1_Timer ilmestyy Code-ikkunaan.

9 Kirjoita seuraava lause:

```
Label1.Caption = Time
```

Tämä lause lukee nykyisen ajan järjestelmäkellosta ja sijoittaa sen Label1-objektin Caption-ominaisuuteen. Ohjelmassa tarvitaan vain yksi lause, koska ajastimen Interval-ominaisuus määritettiin Properties-ikkunassa. Ajastinobjekti huolehtii lopusta.

Käynnistä-painike

10 Sulje Code-ikkuna ja käynnistä ohjelma napsauttamalla työkalurivin Käynnistä-painiketta.

Kello näyttää tällaiselta:

11 Tarkkaile kelloa vähän aikaa.

Visual Basic päivittää ajan sekunnin välein.

12 Sulje kelloikkuna napsauttamalla sen otsikkorivin sulkemispainiketta.

Ohjelma voidaan lopettaa napsauttamalla sulkemispainiketta Lopeta-painikkeen sijasta. Käyttäjä valitsee varmaan tämän menetelmän, jos kello toimii erillisenä ohjelmana Visual Basic -ympäristön ulkopuolella.

Tallenna projekti -painike

13 Napsauta Tallenna projekti -painiketta ja tallenna lomake ja projekti nimellä **MyDigClock**.

Kello on varsin kätevä, ja haluat ehkä kääntää **MyDigClock**-ohjelman suoritettavaksi tiedostoksi, jotta voit käynnistää sen milloin vain. Voit muokata kelloa lisäämällä kuvia, tekstiä ja värejä.

Lisätietoja: Aikarajan määrittäminen ajastinobjektilla

Toinen ajastinobjektin kiinnostava sovellus on tilanne, jossa objektin pitää odottaa tietty aika ja sitten käynnistää tai lopettaa toiminta. Ajastin toimii kuin munakello — määritä Interval-ominaisuudella haluamasi viivästys ja käynnistä kello asettamalla Enabled-ominaisuudeksi True.

Seuraavassa tehtävässä näet, miten ajastimella määritetään aikaraja salasanan kirjoittamiselle. Ajastin lopettaa ohjelman suorituksen, ellei oikeaa salasanaa anneta 15 sekunnin kuluessa. (Tavallisesti tällainen ohjelma toimisi suuremman sovelluksen osana.) Samalla menetelmällä voisit tuoda näyttöön tervetuliais- tai tekijänoikeusilmoituksen tai toistaa tehtävän tietyin välein, esimerkiksi tallentaa tiedoston joka 10. minuutti.

Salasanan aikarajan asettaminen

1 Valitse File-valikosta komento New Project ja napsauta OK.

Tekstiruutu-työkalu

2 Pienennä lomake suunnilleen tavanomaisen tekstiruudun kokoiseksi.

3 Napsauta Tekstiruutu-työkalua.

4 Piirrä lomakkeen keskelle tekstiruutu.

Teksti-työkalu

5 Napsauta Teksti-työkalua ja lisää tekstiruudun yläpuolelle pitkä tekstiobjekti.

6 Napsauta Komentopainike-työkalua ja piirrä tekstiruudun alapuolelle komentopainike.

Komentopainike-työkalu

7 Napsauta Ajastin-työkalua.

8 Lisää lomakkeen vasempaan alakulmaan ajastinobjekti.

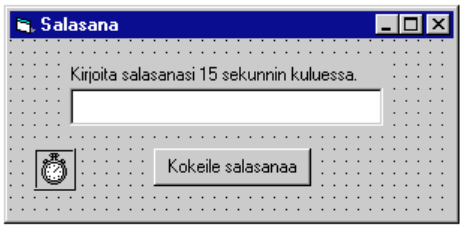
Ajastin-työkalu

9 Määritä ohjelman objektien ominaisuudet seuraavan sivun taulukon mukaisesti.

Objekti	Ominaisuus	Asetus
Text1	Text	(Tyhjä)
	PasswordChar	*
Label1	Caption	”Kirjoita salasanasi 15 sekunnin kuluessa.”
Command1	Caption	”Kokeile salasanaa”
Timer1	Interval	15000
	Enabled	True
Form1	Caption	”Salasana”

PasswordChar-ominaisuuden asetus merkitsee sitä, että ohjelma näyttää tekstiruudussa tähtiä (*), kun käyttäjä kirjoittaa salasanan. Koska Interval-ominaisuus on 15000, käyttäjällä on 15 sekuntia aikaa kirjoittaa salasana ja napsauttaa Kokeile salasanaa -painiketta. Koska Enabled-ominaisuus on True (oletusarvo), ajastin käynnistyy ohjelman käynnistyttyä. Voit myös määrittää ominaisuudeksi False ja vaihtaa sen tapahtuma-aliohjelmassa, jos ajastinta tarvitaan vasta myöhemmin.

Lomakkeen pitäisi näyttää tällaiselta:



10 Kaksoisnapsauta lomakkeen ajastinobjektia ja kirjoita seuraavat lauseet:

```
MsgBox ("Aika ikävä kyllä loppui.")
End
```

Ensimmäinen lause näyttää ilmoituksen ajan loppumisesta ja toinen lopettaa ohjelman. Visual Basic suorittaa tämän tapahtuma-aliohjelman, ellei oikeaa salasanaa anneta 15 sekunnin kuluessa.

11 Valitse Code-ikkunan objektiluettelosta Command1-objekti ja kirjoita tapahtuma-aliohjelmaan Command1_Click seuraavat lauseet:

```
If Text1.Text = "salainen" Then
    Timer1.Enabled = False
    MsgBox ("Tervetuloa  järjestelmään!")
    End
```

```
Else
    MsgBox ("Sori, kaveri, en tunne sinua.")
End If
```

Nämä lauseet tarkistavat, onko tekstiruutuun kirjoitettu salasana ”salainen”. Jos on, aliohjelma poistaa ajastimen käytöstä, näyttää tervetulotoivotuksen ja lopettaa ohjelman. (Hyödyllisempi ohjelma jatkaisi toimintaansa eikä lopettaisi tähän.) Ellei annettu salasana ole oikea, ohjelma ilmoittaa tästä ja antaa käyttäjälle uuden mahdollisuuden kirjoittaa salasana. Hänellä on jälleen vain 15 sekuntia aikaa.

12 Sulje Code-ikkuna ja käynnistä ohjelma napsauttamalla Käynnistä-painiketta.

Käynnistä-painike

Ohjelma käynnistyy, ja 15 sekunnin ajastin alkaa tikittää.

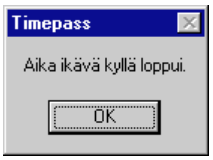
13 Kirjoita tekstiruutuun **avaa** ja napsauta Kokeile salasanaa -painiketta.

Näyttöön tulee seuraava ilmoitusikkuna osoitukseksi siitä, että salasana ei kelpaa:

Valmis Time-Pass-ohjelma on kansiossa \Vb5Sbs\Less06.

14 Napsauta OK ja odota kärsivällisesti, kunnes kirjautumisaika on kulunut.

Ohjelma näyttää tällaisen ilmoituksen:



15 Lopeta ohjelma napsauttamalla OK.

Visual Basic -ohjelmointiympäristö palaa näkyviin.

16 Napsauta Tallenna projekti -painiketta ja tallenna lomake ja projekti nimellä **MyTimePass**.

Tallenna projekti -painike

Jos haluat harjoitella ohjelmointitaitojasi

➤ Lue asiakirja Review & Practice 2, joka on tallennettu kansion \Vb5Sbs\Review tiedostoon Rev2.doc. Se sisältää vaiheittain eteneviä tehtäviä, joilla voit testata ohjelmointitaitosi ja kerrata lukujen 4–6 sisällön. Tehtävissä muodostetaan ohjelma, joka muistuttaa kokouksen alkamisesta.

Jos haluat jatkaa seuraavaan harjoitukseen

➤ Jätä Visual Basic käyntiin ja siirry harjoitukseen 7.

Jos haluat lopettaa Visual Basicin

➤ Valitse File-valikosta komento Exit.

Jos näyttöön tulee Save-valintaikkuna, napsauta Yes.

Harjoituksen tiivistelmä

Tehtävä	Ohje

Suorittaa joukko ohjelmalauseita tietty määrä kertoja	Kirjoita lauseet silmukkaan For- ja Next-lauseiden väliin. Esimerkki:
	`For i = 1 To 10`
	`  MsgBox ("Painoit jo OK!")`
	`Next i`

Näyttää lomakkeessa yksi tai useita tekstirivejä	Käytä Print-metodia. Esimerkki:
	`For Cnt = 1 To 5`
	`  Print "Laskuri on "; Cnt`
	`  Print "Hyvin menee."`
	`Next Cnt`

Käyttää lauseissa tiettyjä juoksevia numeroita	Kirjoita lauseeet For…Next-silmukkaan ja määritä viimeinen numero ja askelarvo To- ja Step-avainsanoilla. Esimerkki:
	`For i = 2 To 8 Step 2`
	`  Print i; "...";`
	`Next i`
	`Print "Mikä näistä on paras?"`

Tehtävä	Ohje	Painike

Keskeyttää For…Next-silmukka	Käytä Exit For -lausetta. Esimerkki:
	`For i = 1 To 10`
	`InpName = InputBox("Nimi?")`
	`If InpName = "Brutus" Then Exit For`
	`Print InpName`
	`Next i`	

Toistaa joukkoa ohjelmalauseita niin kauan kuin annettu ehto on tosi	Kirjoita toistettavat lauseet Do- ja Loop-lauseiden väliin. Esimerkki:
	`Do While Query <> "Joo"`
	`Query = InputBox("Brutus?")`
	`If Query <> "Joo" Then Print Query Loop`	

Välttää loputon Do-silmukka	Varmista, että silmukan lopetusehto saa aikanaan arvon False.	

Toistaa silmukkaa, kunnes annettu ehto on tosi	Käytä Do-silmukassa Until-avainsanaa. Esimerkki:
	`Do`
	`GiveIn = InputBox("Sano 'Muikku'")`
	`Loop Until GiveIn = "Muikku"`	

Toistaa lauseita tietty aika	Käytä ajastin- eli Timer-objektia.	
Lisätä ohjelman nimi sen ikkunan otsikkoriville	Sijoita Form1-objektiin haluamasi nimi.	

Kun tarvitset tietoja aiheesta	Valitse Visual Basicin Help-valikosta Microsoft Visual Basic Help Topics, napsauta Index-välilehteä ja...
For…Next-silmukat	Kirjoita **For statement**.
Do-silmukat	Kirjoita **Do**.
Print-metodi	Kirjoita **Print method**.
Ajastinobjektit	Kirjoita **timer**.

Tietoja seuraavasta harjoituksesta

Harjoituksessa 7 opit lisää ohjelman käyttöliittymän muodostamisesta. Esitän siinä, miten voit lisätä ohjelmaan lomakkeita, lähettää ohjelman tuottamat tiedot tulostimeen ja opettaa ohjelman reagoimaan odottamattomiin tapahtumiin käyttämällä tapahtumankäsittelijöitä.

Osa

3

Käyttöliittymän luominen

Harjoitus 7
Lomakkeet, tulostimet ja virheenkäsittelijät 187

Harjoitus 8
Kuvioiden ja erikoistehosteiden lisääminen 213

Harjoitus

7

Lomakkeet, tulostimet ja virheenkäsittelijät

Arvioitu aika
45 min.

Tässä harjoituksessa opit

- lisäämään ohjelmaan uusia lomakkeita
- lähettämään tietoja tulostimeen
- hoitamaan ajonaikaiset virheet virheenkäsittelijöillä.

Osassa 2 opit, miten Visual Basicin lauseista, funktioista ja ohjausrakenteista kootaan hyödyllinen ohjelma. Osassa 3 keskitymme jälleen käyttöliittymään ja katsomme, miten saadaan aikaan vaikuttavia tehosteita ja luotettavia sovelluksia. Tässä harjoituksessa opit lisäämään käyttöliittymään uusia lomakkeita syöttö- ja tulostustietojen tai erikoisilmoitusten käsittelemistä varten. Opit myös lähettämään ohjelman tulokset tulostimeen ja varautumaan odottamattomiin tuloksiin käyttämällä virheenkäsittelijöitä.

Uusien lomakkeiden lisääminen ohjelmaan

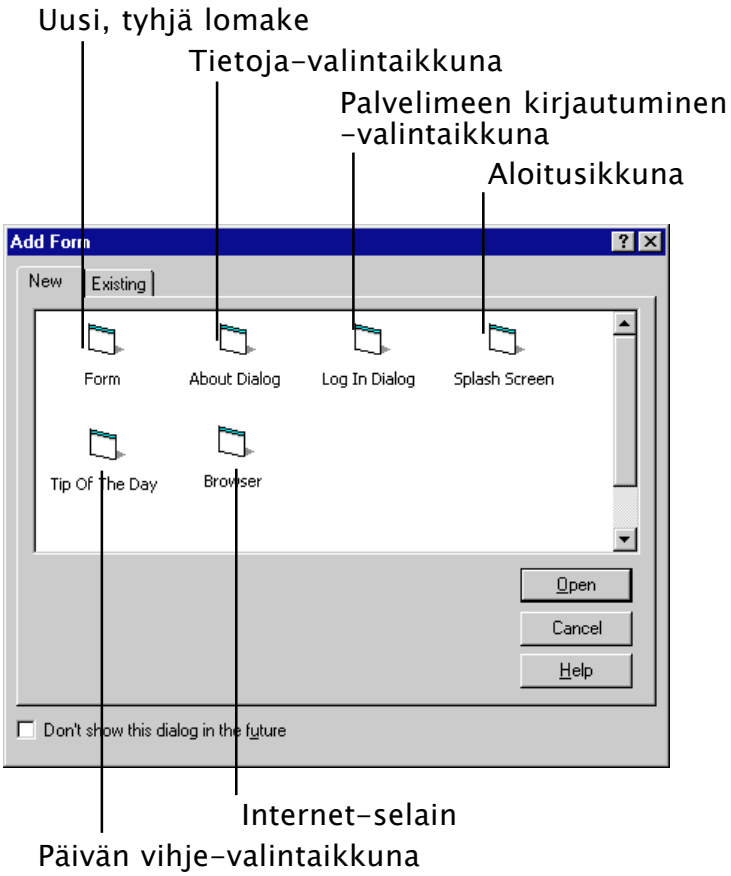
Jokaisella lomakkeella on yksikäsitteinen nimi ja omat objektinsa, ominaisuutensa ja tapahtuma-aliohjelmansa.

Tähän mennessä kirjoittamasi ohjelmat ovat käyttäneet tietojen syöttämiseen ja tulostamiseen vain yhtä lomaketta. Monesti yksi lomake riittääkin viestimiseen käyttäjän kanssa. Jos kuitenkin haluat näyttää ohjelmassa monta ikkunaa, lisää siihen yksi tai useita lomakkeita. Jokainen uusi lomake muodostaa objektin ja sisältää omat objektinsa, ominaisuutensa ja tapahtuma-aliohjelmansa. Ohjelman ensimmäisen lomakkeen nimi on Form1, toisen Form2 ja niin edelleen. Seuraavan sivun taulukossa luetellaan tapoja käyttää ohjelman lisälomakkeita.

Lomake	Kuvaus
Aloitusikkuna	Ikkuna, joka näyttää tervetulotoivotuksen, kuvan tai tekijänoikeustiedot ohjelman käynnistyttyä.
Ohjelman käyttöohjeet	Ikkuna, joka sisältää ohjeita tai vihjeitä ohjelman käyttämisestä.
Valintaikkuna	Mukautettu valintaikkuna, joka lukee ja näyttää tietoja.
Asiakirjan sisältö ja kuvat	Ikkuna, joka näyttää yhden tai usean ohjelmassa käytetyn tiedoston sisällön.

Tyhjät tai valmiit lomakkeet

Voit lisätä uuden lomakkeen valitsemalla Project-valikosta komennon Add Form. Näyttöön tulee seuraavan kuvan mukainen valintaikkuna, jossa kysytään luotavan lomakkeen tyyppi. Voit luoda uuden, tyhjän lomakkeen tai puolivalmiin, tiettyyn tehtävään suunnitellun lomakkeen.



Lomakkeiden käyttö

Lomakkeet voivat olla moodisia tai moodittomia.

Visual Basicissa lomakkeiden käyttö on erittäin joustavaa. Voit tuoda ohjelman kaikki lomakkeet näkyviin samanaikaisesti tai ladata ja poistaa niitä tarpeen mukaan. Jos näytät useita lomakkeita kerralla, voit antaa käyttäjän siirtyä lomakkeesta toiseen tai ohjata järjestystä, jossa lomakkeita käsitellään. Lomaketta, jota on käytettävä sen ollessa näytössä, sanotaan moodiseksi — käyttäjä ei voi siirtyä siitä pois ennen kuin hän on napsauttanut OK tai Peruuta (Cancel) tai muulla tavoin sulkenut sen.

Lomaketta, josta käyttäjä voi siirtyä pois, sanotaan moodittomaksi. Useimmat Windows-sovellukset käyttävät tietojen näyttämiseen moodittomia lomakkeita, koska nämä ovat joustavampia. Siksi uusi lomake on oletuksen mukaan mooditon. Voit määrittää lomakkeen ominaisuudet muista lomakkeista riippumatta, esimerkiksi nimen, koon, reunaviivan tyylin, edusta- ja taustavärit, fontin ja taustakuvan.

Lomakelauseet ohjelmassa

Kun olet luonut lomakkeen ohjelmointiympäristössä, voit ladata sen muistiin ja käsitellä sitä tapahtuma-aliohjelmien lauseilla. Lause, jolla uusi lomake ladataan muistiin, on tällainen:

```
Load lomake
```

Lomake on ladattavan lomakkeen nimi. Esimerkiksi lause

```
Load Form2
```

Load-lause lataa uuden lomakkeen muistiin.

lataa muistiin ohjelman toisen lomakkeen. Kun lomake on ladattu, voit viitata siihen missä tahansa tapahtuma-aliohjelmassa ja käsitellä kaikkia siihen liittämiäsi ominaisuuksia ja metodeja. Jos esimerkiksi haluat määrittää ohjelman toisen lomakkeen Caption-ominaisuudeksi ”Lajittelun tulokset”, voisit kirjoittaa mihin tahansa tapahtuma-aliohjelmaan tällaisen lauseen:

```
Form2.Caption = ”Lajittelun  tulokset”
```

Show-metodi näyttää ladatun lomakkeen.

Kun olet valmis näyttämään ladatun lomakkeen, kutsu sitä Show-metodilla ja määritä, onko se moodinen vai mooditon. Show-metodin kutsu on tällainen:

```
lomake.Show moodi
```

Uusi lomake on oletuksen mukaan mooditon.

Lomake on lomakkeen nimi ja *moodi* on 0 eli mooditon (oletus) tai 1 eli moodinen. Jos esimerkiksi haluat näyttää lomakkeen Form2 moodittomana, kutsu Show-metodia määrittämättä *moodia*:

```
Form2.Show
```

Jos haluat näyttää lomakkeen Form2 moodisena, kirjoita tällainen lause:

```
Form2.Show 1
```

HUOMAA Jos Show-metodia kutsutaan ennen kuin Load-lause on suoritettu, Visual Basic lataa ja näyttää määritetyn lomakkeen automaattisesti. Visual Basicissa on erillinen Load-lause siksi, että ohjelma voi ladata lomakkeet muistiin etukäteen ja Show-metodi toimii nopeasti ja käyttäjät eivät huomaa mitään viivästyksiä. Lomakkeet kannattaa ladata ennakolta, varsinkin jos niissä on useita objekteja tai kuvia.

Lomakkeiden piilottaminen ja poistaminen

Hide-metodi muuttaa lomakkeen näkymättömäksi.

Unload-lause poistaa lomakkeen muistista.

Voit piilottaa lomakkeita Hide-metodilla ja poistaa niitä muistista Unload-lauseella. Nämä avainsanat ovat vastakohtia Show- ja Load-avainsanoille. Kun lomake on piilotettu, se on näkymätön mutta edelleen muistissa myöhempää käyttöä varten. Lomakkeen piilottaminen merkitsee samaa kuin sen muuttaminen näkymättömäksi Visible-ominaisuuden avulla. Kun lomake poistetaan muistista, sen objektille ja kuville varattu muistitila vapautuu. Lomakkeen tapahtuma-aliohjelmat sen sijaan ovat aina muistissa. Voit piilottaa lomakkeen Form2 ja poistaa sen muistista Hide- ja Unload-lauseilla näin:

```
Form2.Hide
Unload Form2
```

TÄRKEÄÄ Kun olet poistanut lomakkeen muistista, sen ajonaikaiset arvot ja ominaisuudet ovat kadonneet. Jos lataat sen uudelleen, se sisältää ohjelmalauseilla määritetyt alkuperäiset arvot.

Lomakkeiden pienentäminen

Voit pienentää lomakkeen (siirtää sen tehtäväpalkkiin) tai suurentaa lomakkeen (laajentaa sen näytön täyttäväksi) käyttämällä WindowState-ominaisuutta. Esimerkiksi seuraava lause pienentäisi ohjelman lomakkeen Form1:

```
Form1.WindowState = 1
```

Suurenna lomake Form1 tällaisella lauseella:

```
Form1.WindowState = 2
```

Palauta lomake Form1 normaalitilaansa tällaisella lauseella:

```
Form1.WindowState = 0
```

Oheisen CD-tietolevyn harjoituksessa Part 2 Review & Practice esitellään WindowState-ominaisuuden toimintaa.

Valmiiden lomakkeiden lisääminen ohjelmaan

Lomakkeita on mahdollista käyttää uudelleen muissa ohjelmointiprojekteissa. Siksi tässä kirjassa kuvatut lomakkeet on tallennettu erillisiin .frm-tiedostoihin. Kun haluat lisätä aiemmin tehdyn lomakkeen projektiin, valitse Visual Basicin Project-valikosta komento Add Form ja napsauta Existing-välilehteä. Näyttöön tulee valintaikkuna, jossa on lueteltu Visual Basicin kotikansion (C:\Program Files\DevStudio\VB) kaikki lomakkeet. Lisää lomake kaksoisnapsauttamalla sen nimeä. Visual Basic liittää lomakkeen projektiin, ja voit katsella sitä ja muokata sen tapahtuma-aliohjelmia napsauttamalla Project-ikkunan Näytä objekti- ja Näytä lauseet -painikkeita. Voit määrittää ohjelman aloituslomakkeen (ensimmäiseksi ladattavan lomakkeen) valitsemalla Project-valikosta komennon Properties, napsauttamalla General-välilehteä ja valitsemalla lomakkeen Startup Object -luettelosta. Jos muokkaat aiemmin tehtyä lomaketta uudessa projektissa, sinun kannattaa tallentaa se toisella nimellä, sillä muuten toiset samaa lomaketta käyttävät ohjelmat eivät enää toimi oikein.

Usean lomakkeen käsittely: Italian-ohjelma

Seuraavaksi esittelen sanakirjaohjelman nimeltä Italian, joka havainnollistaa kuvan ja tekstin näyttämistä toisessa lomakkeessa. Ohjelma esittää italiankielisten sanojen englanninkieliset käännökset alkuaan MsgBox-funktiolla, mutta sinun pitää muokata sitä niin, että se näyttää tiedot toisessa lomakkeessa.

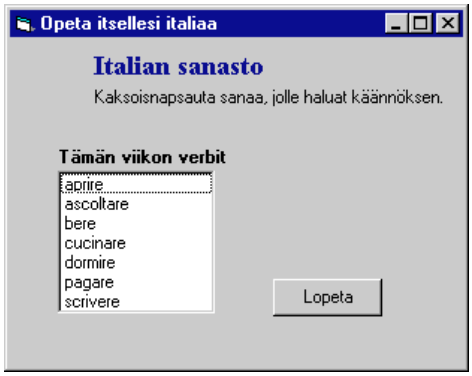
Italian-ohjelman suorittaminen

1. Käynnistä Visual Basic.
2. Napsauta työkalurivin Avaa projekti -painiketta.
3. Avaa Italian-projekti kansiosta \Vb5Sbs\Less07.
4. Ellei Opeta itsellesi italiaa -lomake avaudu, valitse Project-ikkunasta Italian-lomake ja napsauta Näytä objekti -painiketta.

   Ohjelman käyttöliittymä tulee näkyviin. Kokeile ohjelmaa.
5. Napsauta työkalurivin Käynnistä-painiketta.

   Lomakkeen luetteloruudussa näkyy joukko italian kielen verbejä. Kaksoisnapsauta verbiä, niin ohjelma näyttää sen käännöksen.

Käynnistä-painike



6 Kaksoisnapsauta verbiä *dormire.*

Verbin käännös näkyy ilmoitusikkunassa:

MsgBox-funktio näyttää tuloksen käyttökelpoisessa, mutta aika pelkistetyssä muodossa.

7 Sulje ilmoitusikkuna napsauttamalla OK.

Ilmoitusikkuna sulkeutuu, ja päälomake palaa näkyviin.

8 Lopeta ohjelma napsauttamalla Lopeta.

Seuraava tehtäväsi on poistaa MsgBox-funktion kutsu ja korvata se uudella lomakkeella, joka näyttää tiedot. Lisää lomake ohjelmaan napsauttamalla työkalurivin Lomake-painiketta.

Toisen lomakkeen lisääminen ohjelmaan

1 Valitse Project-valikosta komento Add Form.

Add Form -valintaikkuna avautuu. Sen New-välilehdessä näkyy kuusi valmiiksi määritettyä lomaketta.

2 Lisää projektiin uusi, tyhjä lomake napsauttamalla Open.

Ohjelmointiympäristöön ilmestyy tyhjä lomake nimeltä Form2.

3 Muuta toinen lomake pienen valintaikkunan kokoiseksi ja muotoiseksi.

Muokkaa Form2-ikkunaa eikä lomakkeen sisältävää Project1-ikkunaa.

Kuva-työkalu

Teksti-työkalu

Tekstiruutu-työkalu

Komentopainike-työkalu

4 Napsauta Kuva-työkalua ja piirrä lomakkeen vasempaan reunaan pienehkö kuvakehys.

Tähän kuvakehykseen sijoitetaan Italian lipun bittikarttakuva.

5 Napsauta Teksti-työkalua ja lisää lomakkeen yläreunan keskelle teksti-objekti.

6 Napsauta Tekstiruutu-työkalua ja lisää lomakkeen keskelle, tekstiobjektin alapuolelle, suuri tekstiruutu.

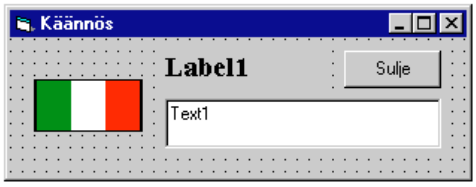
7 Napsauta Komentopainike-työkalua ja piirrä lomakkeen oikeaan reunaan komentopainike.

Määritä uuden lomakkeen objektien ominaisuudet näin:

Objekti	Ominaisuus	Asetus
Image1	Stretch Picture	True ”c:\vb5sbs\less07\flgitaly.ico”
Label1	Font	Times New Roman, Bold, 14 pistettä
Text1	TabStop	False
Command1	Caption	”Sulje”
Form2	Caption	”Käännös”

HUOMAA Kun tekstiruudun TabSTop-ominaisuudeksi annetaan False, käyttäjä ei voi valita tekstiruutua painamalla SARKAIN-NÄPPÄINTÄ. Ellet määritä ominaisuutta näin, kohdistin vilkkuu tekstiruudussa lomakkeen avauduttua.

Määritettyäsi ominaisuudet lomakkeen pitäisi näyttää tällaiselta:



Tallenna uusi projekti ja uusi lomake. Kuten tiedät, Visual Basic edellyttää, että jokainen lomake tallennetaan omaan tiedostoonsa. Usea projekti voi kuitenkin käyttää lomaketta. Ennen kuin tallennat Form2-lomakkeen, varmista, että se on ohjelmointiympäristössä aktiivinen eli valittu (sen pitäisi olla valittu nyt).

8. Valitse File-valikosta komento Save Form2 As.

   Save File As -valintaikkuna avautuu.

9. Tallenna Form2-lomake kansioon \Vb5Sbs\Less07 nimellä **MyDef.frm**.

   Visual Basic tallentaa toisen lomakkeen ja rekisteröi sen Project-ikkunaan. Voit siirtyä lomakkeesta toiseen napsauttamalla lomaketta tai valitsemalla sen nimen Project-ikkunasta ja napsauttamalla Näytä objekti -painiketta.

10. Napsauta Form1-lomaketta ja tallenna se valitsemalla File-valikosta komento Save Italian.frm As. Kirjoita nimi **MyWordList.frm** ja paina ENTER.

    Nyt Form1-lomakkeeseen tekemäsi muutokset eivät vaikuta Italian-projektiin.

11. Valitse File-valikosta komento Save Project As ja tallenna projekti nimellä **MyItalian2.vbp**.

Muokkaa seuraavaksi tapahtuma-aliohjelmaa Text1_DblClick niin, että se näyttää uuden lomakkeen.

Toisen lomakkeen näyttäminen tapahtuma-aliohjelmalla

1. Napsauta ensimmäistä lomaketta ja kaksoisnapsauta sen List1-objektia.

   Tapahtuma-aliohjelma List1_DblClick ilmestyy Code-ikkunaan. Olen kirjoittanut siihen Select Case -rakenteen,joka näyttää valitun italiankielisen sanan käännöksen MsgBox-funktiolla. Rakenne selvittää, minkä vaihtoehdon käyttäjä valitsi, ja sijoittaa asianmukaisen käännöksen Def-muuttujaan. Voit tutkia lauseita vähän aikaa ennen kuin jatkat.

2. Vieritä Code-ikkunaa alaspäin, jotta näet tapahtuma-aliohjelman lopun.

   Seuraava MsgBox-funktion kutsu tulee esiin:

   ```
   MsgBox (Def), , List1.Text
   ```

3. Poista lause ja kirjoita sen tilalle seuraavat lauseet:

   ```
   Load Form2
   Form2.Label1 = List1.Text
   Form2.Text1 = Def
   Form2.Show
   ```

Objektin nimi Form2 tarkoittaa ohjelman uutta lomaketta.

Ensimmäinen lause lataa Form2-lomakkeen muistiin. (Voisit myös esiladata lomakkeen Form_Load-aliohjelmassa.) Kun lomake on muistissa, voit muuttaa sen ominaisuuksia ja valmistella sen näyttämistä varten. Toinen lause sijoittaa valitun italiankielisen sanan kopion Form2-lomakkeen ensimmäiseen tekstiin. Kolmas lause sijoittaa Def-muuttujan arvon (sanan englanninkielisen käännöksen) uuden lomakkeen tekstiruutuun. Tekstiruutu tarvitaan siksi, että sanan käännös voi olla pitkä tai sisältää useita vaihtoehtoja. Jos tekstiä on paljon, tekstiruudun reunaan ilmestyy vierityspalkki, jota käyttäen saat näkyviin haluamasi osan tekstistä. Lopuksi Show-metodi näyttää valmiin lomakkeen.

Lisää Form2-lomakkeen Sulje-painikkeeseen lause, joka sulkee lomakkeen, kun käyttäjä ei enää tarvitse sitä.

4 Sulje Code-ikkuna, napsauta Form2-lomaketta (tai näytä se käyttämällä Project-ikkunaa) ja kaksoisnapsauta Sulje-painiketta.

Tapahtuma-aliohjelma Command1_Click ilmestyy Code-ikkunaan. Se liittyy ensimmäiseen painikkeeseen Form2-lomakkeessa — ei Form1-lomakkeessa. Eri lomakkeiden objekteilla voi olla samat nimet, ja Visual Basic pystyy vaivatta pitämään ne erillään. Jos samantapaiset tai samat nimet ovat mielestäsi harhaanjohtavia, muuta niitä Properties-ikkunassa.

Eri lomakkeiden objekteilla voi olla samat nimet.

HUOMAA Jos haluat vaihtaa objektin nimen, valitse objekti lomakkeesta ja muuta sen Name-ominaisuutta Properties-ikkunassa. Name-ominaisuus sisältää nimen, jonka perusteella Visual Basic löytää objektin. Jos vaihdat objektin nimen, muista vaihtaa se kaikkialla ohjelmassa. Objektin nimen pitäisi olla looginen, jotta se helpottaa objektin tehtävän muistamista. Siinä pitäisi olla samanlainen tunnistamista helpottava etuliite kuin muuttujissa. Esimerkiksi komentopainike, jolla ohjelma lopetetaan, voisi olla nimeltään cmdQuit.

5 Kirjoita tapahtuma-aliohjelmaan seuraava lause:

```
Form2.Hide
```

Lause piilottaa Form2-lomakkeen Hide-metodilla, kun käyttäjä napsauttaa Sulje-painiketta. Koska Form2-lomake näytetään moodittomana, käyttäjä voi vapaasti siirtyä Form1- ja Form2-lomakkeiden välillä ohjelman suorituksen aikana. Hän voi sulkea Form2-lomakkeen napsauttamalla Sulje-painiketta.

Hide-metodi piilottaa uuden lomakkeen, kun käyttäjä napsauttaa Sulje.

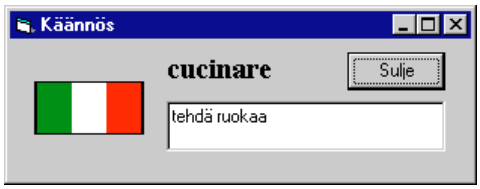
6 Tallenna muokattu projekti napsauttamalla Tallenna projekti -painiketta.

Tallenna projekti -painike

7 Käynnistä ohjelma napsauttamalla Käynnistä-painiketta.

8 Kaksoisnapsauta luettelossa oleva verbiä *cucinare*.

Ohjelma näyttää sanan käännöksen toisessa lomakkeessa seuraavan kuvan mukaisesti:



Kokeile siirtymistä lomakkeesta toiseen.

9 Napsauta ensimmäistä lomaketta ja kaksoisnapsauta sanaa *scrivere.*

Ohjelma näyttää sanan *scrivere* käännöksen (kirjoittaa) toisessa lomakkeessa. Koska lomakkeet ovat moodittomia, voit siirtyä lomakkeesta toiseen milloin haluat.

Valmis Italian2-ohjelma on kansiossa \Vb5Sbs\Less07.

10 Napsauta toisen lomakkeen Sulje-painiketta.

Ohjelma piilottaa lomakkeen.

11 Napsauta ensimmäisen lomakkeen Lopeta-painiketta.

Ohjelma pysähtyy, ja Visual Basic poistaa molemmat lomakkeet muistista. Ohjelmointiympäristö palaa näkyviin.

Ohjelman tietojen lähettäminen tulostimeen

Tulostinobjekti ohjaa tulostusta.

Voit lähettää järjestelmään asennettuun tulostimeen tietoja Print-metodilla. Käytit Print-metodia harjoituksen 6 silmukassa, joka kirjoitti lomakkeeseen tekstiä. Kun haluat lähettää tietoja tulostimeen, käytä tulostin- eli Printer-objektia. Esimerkiksi seuraava lause lähettää Windowsin oletustulostimeen merkkijonon ”Meripoika”:

```
Printer.Print "Meripoika"
```

Voit muuttaa Printer-objektilla eräitä fontin ominaisuuksia ennen tulostusta. Esimerkiksi seuraava lause tulostaa merkkijonon ”Meripoika” 14 pisteen tekstillä:

```
Printer.FontSize = 14
Printer.Print "Meripoika"
```

Printer-objektissa on useita kymmeniä ominaisuuksia ja metodeja, jotka ohjaavat tulostuksen eri piirteitä. Et voi kuitenkaan määrittää Printer-objektin ominaisuuksia Properties-ikkunassa samalla tavoin kuin lomakkeiden ja työkaluilla luotavien objektien ominaisuuksia. Jokainen ominaisuus on asetettava ohjelmalauseella suorituksen aikana.

MDI-lomakkeet eli hierarkkiset ikkunat

Useimmissa ohjelmissasi on todennäköisesti ”kotilomake”, jossa käyttäjä tekee suurimman osan työstään, ja lisäät erityislomakkeita tietojen syöttämistä ja tulostamista varten. Visual Basicilla on mahdollista muodostaa myös *hierarkia* eli erityinen suhde ohjelman sellaisten lomakkeiden välille, jotka toimivat parhaiten ryhmänä. Näitä erityislomakkeita sanotaan MDI (Multiple Document Interface) -lomakkeiksi, ja ne jakautuvat ylilomakkeisiin ja alilomakkeisiin. MDI-ylilomake muodostetaan valitsemalla Project-valikosta komento Add MDI Form ja MDI-alilomake valitsemalla Project-valikosta komento Add Form ja asettamalla lomakkeen MDIChild-ominaisuudeksi True. MDI-lomakkeet toimivat suorituksen aikana samoin kuin tavalliset lomakkeet seuraavia poikkeuksia lukuun ottamatta:

- Jokainen alilomake näytetään yli-ikkunansa rajoittamalla alueella.
- Kun alilomake pienennetään, se kutistuu pieneksi otsikkoriviksi MDI-lomakkeeseen sen sijaan, että se näkyisi painikkeena tehtäväpalkissa.
- Kun ylilomake pienennetään, se ja sen kaikki alilomakkeet näkyvät yhtenä painikkeena tehtäväpalkissa.
- Kaikki alilomakkeen valikot näkyvät ylilomakkeen valikkorivillä. Kun alilomake suurennetaan, sen otsikko näkyy ylilomakkeen otsikkorivillä.
- Voit näyttää ylilomakkeen kaikki alilomakkeet määrittämällä ylilomakkeen AutoShowChildren-ominaisuudeksi True.

Yli- ja alilomakkeet ovat hyödyllisiä varsinkin niin sanotuissa *asiakirjakeskeisissä sovelluksissa*, joissa on monta ikkunaa asiakirjojen tarkastelemista ja muokkaamista varten. Jos haluat tietää lisää MDI-lomakkeista ja niiden sovelluksista, etsi Visual Basicin ohjeista hakusanaa *forms, MDI*.

Form_Load-aliohjelma on hyvä paikka vakioasetuksille, joita pitää käyttää aina ohjelman käynnistyttyä. Monet Printer-objektin ominnaisuudet ja metodit ovat kuitenkin samanlaisia kuin avainsanat, joita olet jo käyttänyt lomakkeissa ja työkaluruudun työkaluilla luomissasi objekteissa. Seuraavan sivun taulukoissa luetellaan hyödyllisiä Printer-objektin ominaisuuksia ja metodeja.

HUOMAA Löydät Printer-objektin ominaisuuksien ja metodien täydellisen luettelon etsimällä Visual Basicin ohjeista hakusanaa *Printer object*. Ominaisuuksista saat myös selville käytettävissä olevat tulostimen toiminnot.

Ominaisuus	Kuvaus
FontName	Määrittää tekstin fontin nimen.
FontSize	Määrittää tekstin fontin koon.
FontBold	Arvolla True muuttaa tekstin lihavoiduksi.
FontItalic	Arvolla True muuttaa tekstin kursivoiduksi.
Page	Sisältää tulostettavana olevan sivun numeron.

Metodi	Kuvaus
Print	Tulostaa määritetyn tekstin.
NewPage	Aloittaa tulostustyössä uuden sivun.
EndDoc	Osoittaa tulostustyön loppuneen.
KillDoc	Keskeyttää nykyisen tulostustyön.

Seuraavaksi sinun on tarkoitus lisätä aiemmin tässä luvussa tekemääsi MyItalian2-ohjelmaan yksinkertainen tulostimen tuki. Muuta tekstin tyyliä FontName-, FontSize- ja FontBold-ominaisuuksilla, lähetä määritykset tulostimeen Print-metodilla ja ilmoita tulostustyön loppumisesta EndDoc-metodilla.

Tulostimen tuen lisääminen MyItalian2-ohjelmaan

Valmis MyItalian2-projekti on kansiossa \Vb5Sbs\Less07.

1. Avaa MyItalian2-projektitiedosto, ellei se ole jo avoinna.

   Ellet kirjoittanut tätä ohjelmaa, lataa kiintolevyltä tiedosto Italian2.vbp.

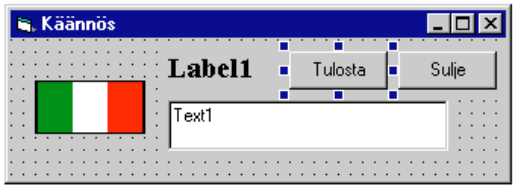
2. Näytä projektin toinen lomake (MyDef.frm tai Def.frm, jos käytät tiedostoa Italian2.vbp).

   Tämä lomake näyttää englanninkielisen käännöksen italiankieliselle sanalle, jonka käyttäjä valitsee luetteloruudusta kaksoisnapsauttamalla. Lisää tähän lomakkeeseen Tulosta-painike, jotta käyttäjä voi tulostaa käännöksen paperille.

Komentopainike-työkalu

3. Napsauta Komentopainike-työkalua ja lisää komentopainike Sulje-painikkeen vasemmalle puolelle.

   Joudut ehkä tekemään painikkeelle tilaa leventämällä lomaketta ja siirtämällä Sulje-painiketta oikealle tai lyhentämällä Label1-objektia. Muokatun lomakkeen pitäisi näyttää tällaiselta:



4 Vaihda Properties-ikkunassa painikkeen tekstiksi ”Tulosta”.

5 Kaksoisnapsauta uutta Print-painiketta, jotta pääset muokkaamaan sen tapahtuma-aliohjelmaa.

Tapahtuma-aliohjelma Command2_Click ilmestyy Code-ikkunaan.

6 Kirjoita aliohjelmaan seuraavat lauseet:

```
Printer.Print ""
Printer.FontName = "Arial"
Printer.FontSize = 18
Printer.FontBold = True
Printer.Print Label1.Caption
Printer.FontBold = False
Printer.Print Text1.Text
Printer.EndDoc
```

Aliohjelman lauseet toimivat näin:

- Ensimmäinen lause alustaa Printer-objektin, jotta se on valmis tulostamista varten.
- FontName-ominaisuus valitsee tulostimen fontiksi TrueType-fontin nimeltä Arial.
- FontSize-ominaisuus määrittää fontin kooksi 18 pistettä.
- Kun FontBold-ominaisuudeksi asetetaan True, teksti muuttuu lihavoiduksi.
- Viides ja seitsemäs lause tulostavat italiankielisen sanan ja sen käännöksen Print-metodilla.
- Viimeinen lause päättää tulostustyön ja lähettää sen tulostimeen.

Kun käytät Windowsin mukana toimitettua TrueType-fonttia, ohjelmasi on yhteensopiva useimpien tulostimien kanssa.

HUOMAA Jos käyttäjän pitää voida tulostaa useita sanoja ja niiden käännöksiä samalle riville, sinun kannattaa ehkä lykätä EndDoc-metodin suorittaminen siihen asti, että käyttäjä lopettaa ohjelman napsauttamalla Lopeta.

7 Valitse File-valikosta komento Save MyDef.frm As ja tallenna toinen lomake nimellä **MyPrintFrm.frm**. Näin alkuperäinen MyDef.frm säilyy ennallaan.

8 Valitse File-valikosta komento Save Project As ja tallenna uusi projekti nimellä **MyPrintFrm.vbp**.

Valmis PrintFrm-projekti on kansiossa \Vb5Sbs\Less07.

Koska et tallentanut tiedostoa MyWordList.frm uudella nimellä, sen sisältämää lomaketta ja lauseita käyttää sekäMyPrintFrm.vbp että MyItalian2.vbp. Aina kun teet muutoksen MyWordList.frm-tiedostoon (lomakkeeseen tai tapahtuma-aliohjelmiin), muutos vaikuttaa kumpaankin projektiin.

MyPrintFrm-ohjelman suorittaminen

Jos järjestelmääsi on asennettu tulostin, käynnistä ohjelma. Se käyttää Windowsin Kirjoittimet-kansiossa (Printers) määritettyä oletustulostinta, joten laite voi olla paikallistulostin, verkkotulostin tai faksimodeemiohjelma. Varmista, että tulostin on linjalla ja toimintavalmis.

Käynnistä-painike

1 Napsauta työkalurivin Käynnistä-painiketta.

Ohjelma käynnistyy ohjelmointiympäristössä.

2 Kaksoisnapsauta luettelossa näkyvää italiankielistä sanaa *bere*.

Sanan käännös näkyy kuvan mukaisessa lomakkeessa:

3 Tulosta käännös paperille napsauttamalla Tulosta-painiketta.

Visual Basic lähettää asiakirjan oletustulostimeen.

VAROITUS Ellei tulostin ole toimintavalmis, Windows voi palauttaa Visual Basiciin virhekoodin, jota ohjelmasi ei pysty käsittelemään. Tästä saattaa aiheutua ajonaikainen virhe tai ohjelman kaatuminen. Opit käsittelemään levyasemiin, tulostimiin ja muihin laitteisiin liittyvät ajonaikaiset virheet myöhemmin tässä harjoituksessa.

4 Sulje käännöksen sisältävä ikkuna napsauttamalla Sulje ja lopeta ohjelma napsauttamalla Lopeta.

Ohjelman suoritus päättyy, ja ohjelmointiympäristö palaa näkyviin.

Koko lomakkeen tulostaminen PrintForm-metodilla

Sen sijaan, että tulostaisit rivejä yksitellen Print-metodilla, voit lähettää yhden tai usean lomakkeen koko sisällön tulostimeen PrintForm-metodilla. Näin pystyt järjestämään haluamasi tekstin, grafiikan ja käyttölittymän osat lomakkeeseen ja sitten lähettämään koko lomakkeen tulostimeen. Käytä PrintForm-avainsanaa sellaisenaan nykyisen lomakkeen tulostamiseen tai määritä tulostettavan lomakkeen nimi. Jos esimerkiksi haluat tulostaa ohjelman toisen lomakkeen sisällön, kirjoita lause

```
Form2.PrintForm
```

ohjelman mihin tahansa tapahtuma-aliohjelmaan.

Seuraavasta esimerkistä näkyy, miten PrintForm-metodilla tulostetaan lomake, jossa on sekä tekstiä että grafiikkaa. PrintForm-metodi on useimmiten helpoin tapa lähettää kuva tulostettavaksi.

HUOMAA PrintForm-metodi tulostaa lomakkeen näyttösovittimen nykyisten asetusten mukaisella tarkkuudella, joka on tavallisesti 96 pistettä tuumalla.

Tekstin ja grafiikan tulostaminen PrintForm-metodilla

Teksti-työkalu

Komentopainike-työkalu

1 Valitse File-valikosta komento New Project ja luo vakiosovellus napsauttamalla OK.

2 Napsauta Teksti-työkalua ja piirrä keskikokoinen tekstikehys lomakkeen keskiosaan.

3 Napsauta Komentopainike-työkalua ja piirrä komentopainike lomakkeen oikeaan alakulmaan.

4 Määritä ohjelman objektien ominaisuudet näin:

Objekti	Ominaisuus	Asetus
Label1	Caption	"Kuukausiraportti"
	BackStyle	0 - Transparent
	Font	MS Sans Serif, Bold, 14 pistettä
Command1	Caption	"Tulosta tämä!"
Form1	Picture	"c:\vb5sbs\less07\prntout2.wmf"

5 Kaksoisnapsauta Tulosta tämä! -painiketta, jotta sen tapahtuma-aliohjelma tulee näkyviin.

6 Kirjoita seuraava lause:

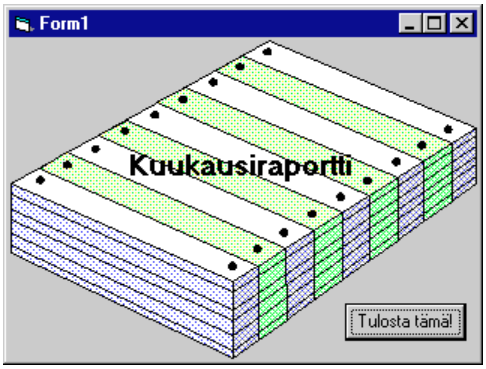
```
Form1.PrintForm
```

Tallenna projekti -painike

7 Tallenna lomake ja projekti napsauttamalla työkalurivin Tallenna projekti -painiketta. Kirjoita sekä lomakkeen että projektin nimeksi **MyPrintWMF**.

8 Käynnistä ohjelma napsauttamalla työkalurivin Käynnistä-painiketta.

Ohjelma näyttää tällaisen kuvan ja tekstin:



9 Tulosta lomakkeen sisältö napsauttamalla Tulosta tämä! -painiketta.

Visual Basic lähettää tulostimeen lomakkeen koko sisällön (tekstin, Windows-metatiedoston ja komentopainikkeen). Tulostuksen aikana näytössä voi olla tällainen Visual Basic -valintaikkuna:

Valmis PrintWMF -projekti on kansiossa \Vb5Sbs\Less07.

Ohjelman tuottama sivu ilmestyy tulostimesta hetken kuluttua.

10 Lopeta ohjelma napsauttamalla lomakkeen Sulje-painiketta.

PrintForm-metodi tulostaa lomakkeen kaikki näkyvät objektit.

TÄRKEÄÄ PrintForm-metodi tulostaa ainoastaan lomakkeen näkyvissä olevat objektit. Jos haluat estää tiettyjen objektien (esimerkiksi tämän esimerkin Tulosta tämä! -painikkeen) tulostumisen, määritä niiden Visible-ominaisuudeksi False ennen kuin kutsut PrintForm-metodia. Kun olet lähettänyt lomakkeen tulostimeen, palauta objektit näkyviin.

Virheisiin varautuminen virheenkäsittelijöillä

Etkö ole vielä törmännyt Visual Basic -ohjelmassa ajonaikaiseen virheeseen? *Ajonaikainen virhe* tai *ohjelman kaatuminen* on ohjelman suorituksen aikana sattuva odottamaton tapahtuma, josta Visual Basic ei pysty toipumaan. Koit ehkä ohjelman kaatumisen ensimmäisen kerran tässä harjoituksessa, kun yritit tulostaa, mutta jokin oli vialla. (Tulostimesta oli paperi lopussa tai katkaistu virta, ja Visual Basic ilmoitti tulostimen virheestä.) Ajonaikainen virhe syntyy, kun Visual Basic ei jostakin syystä pysty suorittamaan ohjelmalausetta loppuun. Syy ei ole siinä, että Visual Basic ei olisi tarpeeksi luotettava, vaan se, että kääntäjälle ei ole kerrottu, mitä sen pitää tehdä jonkin mentyä vikaan.

Virheenkäsittelijä auttaa ohjelmaa toipumaan ajonaikaisista virheistä.

Sinun ei onneksi ole pakko sopeutua satunnaisiin virheisiin, jotka kaatavat ohjelmasi. Voit kirjoittaa Visual Basicilla erityisaliohjelmia eli virheenkäsittelijöitä, jotka reagoivat ajonaikaisiin virheisiin. Virheenkäsittelijä hoitaa ajonaikaisen virheen siten, että se kertoo ohjelmalle, miten sen tulee jatkaa, jos jotakin sen lauseista ei voi suorittaa. Virheenkäsittelijät kirjoitetaan samoihin tapahtuma-aliohjelmiin, joissa mahdollisesti epävakaat lauseet sijaitsevat. Ne huolehtivat ongelmasta käyttämällä erityistä virheenkäsittelyobjektia nimeltä Err. Err-objektilla on Number-ominaisuus, joka ilmaisee virheen ja antaa ohjelmalle mahdollisuuden vastata siihen. Jos virhe johtuu esimerkiksi levykeasemasta, virheenkäsittelijä voi näyttää virheilmoituksen ja estää levykkeen käyttämisen, kunnes käyttäjä on korjannut ongelman.

Virheenkäsittelijöiden käyttötilanteita

Useimmat ajonaikaiset virheet johtuvat ulkoisista tapahtumista.

Voit käyttää virheenkäsittelijöitä missä tahansa tilanteessa, jossa odottamaton tapahtuma saattaisi aiheuttaa ajonaikaisen virheen. Virheenkäsittelijällä hoidetaan tavallisesti ulkoiset tapahtumat, jotka vaikuttavat ohjelmaan, esimerkiksi tapahtumat, joiden syy on verkkoaseman vika, levykeaseman avoin salpa tai linjalta poistettu tulostin. Seuraavan sivun taulukossa on lueteltu ongelmia, joista virheenkäsittelijät voivat huolehtia.

Ongelman syy	Kuvaus
Verkko	Verkkoasema tai resurssi, joka lakkaa yllättäen toimimasta.
Levykeasema	Alustamaton tai väärin alustettu levyke, avoin levykeaseman salpa tai viallinen sektori.
Tulostin	Tulostin, joka ei ole linjalla, josta on loppunut paperi tai joka ei muuten ole käytettävissä.
Ylivuoto	Tulostettavia tai piirrettäviä tietoja on liikaa.
Muistin loppuminen	Windowsissa ei ole tarpeeksi muistitilaa sovellukselle tai resurssille.
Leikepöytä	Ongelmat, jotka johtuvat tietojen siirtämisestä tai Windowsin Leikepöydästä.
Ohjelman logiikka	Kielioppi- ja loogiset virheet, joita kääntäjä ei havainnut ja joita ei huomattu testeissä (esimerkiksi väärin kirjoitettu tiedoston nimi).

Ansan asettaminen: On Error -lause

On Error -lauseella valitaan virheenkäsittelijä.

Ohjelma lause, jolla havaitaan ajonaikainen virhe, on nimeltään On Error. Voit sijoittaa sen tapahtuma-aliohjelmaan heti sen lauseen edelle, jonka arvelet voivan aiheuttaa ongelmia. On Error -lause muodostaa eli ottaa käyttöön tapahtuma-ansan kertomalla Visual Basicille, mistä sen pitää jatkaa ohjelman suoritusta virheen sattuessa. On Error -lauseen muoto on tällainen:

```
On Error GoTo viite
```

Viite on virheenkäsittelijän nimi.

Virheenkäsittelijät sijoitetaan lähelle tapahtuma-aliohjelman loppua, On Error -lauseen jälkeen. Jokaisella virheenkäsittelijällä on tunnistusta varten oma viite, jota seuraa kaksoispiste (:), esimerkiksi ErrorHandler: tai PrinterError:. Virheenkäsittelijässä on yleensä kaksi osaa. Ensimmäinen osa käyttää Err.Number-ominaisuutta ehtorakenteessa (esimerkiksi If...Then tai Select Case) ja näyttää ilmoituksen tai asettaa ominaisuuden virheen tyypin perusteella. Toinen osa on Resume-lause, joka aiheuttaa paluun takaisin ohjelmaan, jotta ohjelman suoritus voi jatkua.

Resume-lause

Resume, Resume Next, ja Resume viite *aiheuttavat paluun ohjelmaan.*

Resume-lauseessa voi olla Resume-avainsana yksinään, Resume Next -avainsanat tai Resume-avainsana ja viite, johon hypätään. Valinta riippuu siitä, mistä ohjelman kohdasta suoritusta pitää jatkaa.

Resume-avainsana jatkaa suoritusta lauseesta, joka aiheutti virheen (toivotaan, että virhe on korjattu tai että sitä ei tapahdu uudelleen). Se on hyvä valinta, jos pyydät käyttäjää korjaamaan ongelman, esimerkiksi sulkemaan levykeaseman salvan tai varmistamaan, että tulostin on toimintavalmis.

Resume Next -avainsanat jatkavat suoritusta virheen aiheuttanutta lausetta *seuraavasta* lauseesta. Käytä sitä, jos virheellinen lause pitää jättää väliin. Voit kirjoittaa Resume-avainsanan jälkeen myös viitteen, johon hypätään. Näin suoritus voi jatkua tapahtuma-aliohjelman mistä tahansa kohdasta. Tavanomainen kohde on aliohjelman viimeinen rivi.

Virheenkäsittelijä levykeasemaa varten

Seuraavassa esimerkissä näytän, miten kirjoitat virheenkäsittelijän, joka selviytyy levykeaseman aiheuttamista virheistä. Lisää virheenkäsittelijä ohjelmaan, joka yrittää lukea Windows-metatiedoston aseman A levykkeellä olevasta tiedostosta. Samalla menetelmällä voit lisätä virheenkäsittelyn mihin tahansa Visual Basic -ohjelmaan; muuta vain virhenumeroita ja ilmoituksia.

HUOMAA Seuraava ohjelma tunnistaa ajonaikaisen virheen sen numeron perusteella (luetaan Err.Number-ominaisuudesta). Jos haluat nähdä virhenumeroiden täydellisen luettelon, etsi Visual Basicin ohjeista hakusanaa *errors, numbers*.

Virheenkäsittelijän kirjoittaminen levykeasemaa varten

1. Avaa DriveErr-projektitiedosto.

   DriveErr-lomake ilmestyy Project-ikkunaan.

2. Ellei lomaketta näy, valitse Project-ikkunasta DriveErr-lomake ja napsauta Näytä objekti -painiketta.

3. Kaksoisnapsauta lomakkeen Tarkista asema -painiketta.

   Tapahtuma-aliohjelma Command1_Click ilmestyy Code-ikkunaan. Se lukee aseman A pääkansiosta Windows-metatiedoston Prntout2.wmf, mutta aiheuttaa virheen, jos tiedostoa ei ole tai levykeaseman salpa on auki.

4. Kirjoita tapahtuma-aliohjelman alkuun tällainen lause:

   ```
   On Error GoTo DiskError
   ```

   Tämä virheenkäsittelijä hoitaa ongelmat, jotka aiheuttaa levykeasema.

   Lause ottaa käyttöön virheenkäsittelijän ja ilmoittaa, mistä Visual Basicin pitää jatkaa suoritusta ajonaikaisen virheen tapahduttua. Lisää aliohjelman loppuun DiskError-virheenkäsittelijä.

5 Siirrä kohdistin LoadPicture-lausetta seuraavalle riville ja kirjoita seuraavat lauseet:

```
Exit Sub  ' lopeta aliohjelma

DiskError:
    If Err.Number = 71 Then  ' jos levy ei ole valmiina
        MsgBox ("Sulje levykeaseman salpa."), , _
          "Levyasema ei ole valmiina"
        Resume
    Else
        MsgBox ("prntout2.wmf ei ole asemassa A:\."), , _
          "Tiedostoa ei löydy"
        Resume StopTrying
    End If
StopTrying:
```

Virheenkäsittelijän If...Then-lauseessa oleva ehtolauseke tutkii, onko Err.Number-ominaisuuden arvo 71 eli virhekoodi, joka ilmaisee, että levykeasema ei toimi. Jos on tapahtunut levykevirhe, ohjelma antaa käyttäjälle mahdollisuuden korjata se (sulkemalla levykeaseman salpa tai asettamalla uusi levyke) ja jatkaa tiedoston lataamista. Resume-avainsana merkitsee sitä, että LoadPicture-funktion kutsu toistetaan.

Jos virhe ei liity levykeasemaan, ohjelma olettaa, että levyke on kunnossa mutta että tiedostoa ei löydy pääkansiosta. Silloin virheenkäsittelijä aiheuttaa siirtymisen StopTrying-viitteeseen, joka on aliohjelman lopussa. Kummassakin tapauksessa virheenkäsittelijä antaa käyttäjälle ilmoituksen ja pysäyttää ohjelman ennen kuin sen suoritus keskeytyy virheeseen. Voit lisätä virheenkäsittelijään uusia ElseIf-lauseita ja virhenumeroita, jotta se osaa antaa käyttäjälle tarkempia ja hyödyllisempiä tietoja levyvirheestä.

Exit Sub –lauseella ohitetaan virheenkäsittelijä aliohjelmassa.

Jos ohjelma ei havaitse levykeongelmia tai käyttäjä korjaa vian, ohjelman suoritus jatkuu, kunnes Exit Sub -lause päättää tapahtuma-aliohjelman. Exit Sub on yleiskäyttöinen lause, jolla voidaan päättää mikä tahansa Visual Basic -aliohjelma ennen End Sub -lausetta. Tässä tapauksessa Exit Sub -lause estää virheenkäsittelijän suorittamisen, kun ohjelma on onistunut lataamaan Windows-metatiedoston.

6 Valitse File-valikosta komento Save DriveErr.frm As ja tallenna lomake nimellä **MyFinalErr**.

7 Valitse File-valikosta komento Save Project As ja tallenna projekti nimellä **MyFinalErr**.

8 Kopioi Windows Resurssienhallinnalla (Explorer) tiedosto Prntout2.wmf kansiosta C:\Vb5Sbs\Less07 asemassa A olevalle levykkeelle (tarvitset tätä vaihetta varten levykkeen).

Muista sijoittaa tiedosto levykkeen pääkansioon (A:\).

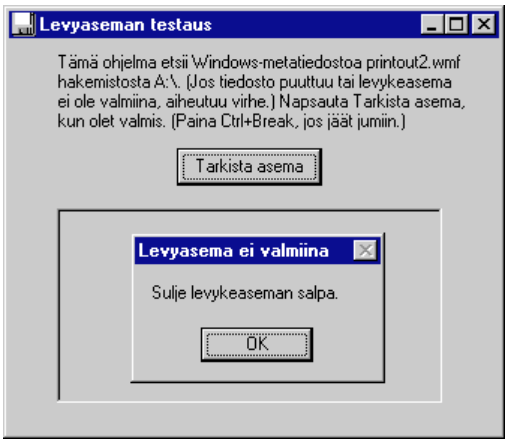
Valmis FinalErr-ohjelma on kansiossa \Vb5Sbs\Less07.

9 Ota levyke asemasta A tai avaa aseman salpa.

10 Käynnistä ohjelma napsauttamalla Käynnistä-painiketta.

11 Napsauta lomakkeen Tarkista asema -painiketta.

Visual Basic aiheuttaa ajonaikaisen virheen, ja virheenkäsittelijä näyttää seuraavan kuvan mukaisen virheilmoituksen:



12 Työnnä Windows-metatiedoston sisältävä levyke asemaan tai sulje aseman salpa.

13 Lopeta virheenkäsittelijän suoritus napsauttamalla OK ja yritä ladata tiedosto uudelleen.

Windows-metatiedosto Prntout2.wmf näkyy kohta lomakkeessa seuraavan kuvan mukaisena:

VIHJE Jos näyttöön tulee jälleen virheilmoitus, ohjelmassa on ilmeisesti jokin muu vika. Kokeile ohjelman suorittamista lause kerrallaan askellustilassa. Virheet on helppo paikantaa käyttämällä pikaseurantalausekkeita, ja samalla näet, miten virheenkäsittelijät toimivat.

14 Lopeta ohjelma napsauttamalla lomakkeen sulkemispainiketta.

Lisätietoja: Muita tapoja käsitellä virheet

Err.Description-ominaisuus sisältää ajon-aikaisen virheen kuvauksen.

Err-objektissa on muutamia muita ominaisuuksia, joita voit käyttää virheen-käsittelijässä lisätietojen näyttämiseen. Err.Description-ominaisuus sisältää Visual Basicin palauttaman virheilmoituksen, jos ajonaikainen virhe on tapahtunut. Voit näyttää sen lisätietona käyttäjälle riippumatta siitä, aiotko käsitellä virheen ohjelmallisesti. Esimerkiksi seuraava virheenkäsittelijä näyttää Description-ominaisuuden avulla virheilmoituksen, jos virhe tapahtuu ladattaessa kuvatiedostoa levykkeeltä:

```
On Error GoTo DiskError
Image1.Picture = LoadPicture("a:\prntout2.wmf")
Exit Sub                    '  lopeta  aliohjelma

DiskError:
MsgBox (Err.Description), , "Latausvirhe"
Resume                      '  yritä LoadPicture-funktiota  uudelleen
```

Näin voit käsitellä levykevirheet, jotka johtuvat alustamattomasta levykkeestä, tiedoston puuttumisesta tai avoimesta levykeaseman salvasta. Virheenkäsittelijässä on Resume-lause, jonka ansiosta ohjelma yrittää ladata tiedoston uudelleen, kun käyttäjä on korjannut ongelman ja napsauttanut ilmoitusikkunassa OK. Kun tiedosto on lopulta ladattu, Exit Sub -lause lopettaa tapahtuma-aliohjelman.

HUOMAA Jos ohjelma juuttuu virhesilmukkaan eikä pysty etene-mään, lopeta se painamalla CTRL+BREAK.

Uudelleenyritysten määrän rajoittaminen

Toinen virheenkäsittelijässä sovellettava menetelmä on yrittää tehtävää muutama kerta ja lopulta ohittaa ongelma, ellei se ratkea. Esimerkiksi seuraava virheenkäsittelijä pitää Retries-muuttujalla lukua virheilmoituksen toistumiskerroista. Jos tiedoston lataaminen epäonnistuu kahdesti, ohjelma jättää latauslauseen väliin.

```
Retries = 0               'alusta   laskurimuuttuja
On Error GoTo DiskError
Image1.Picture = LoadPicture("a:\prntout2.wmf")
Exit Sub                  'lopeta   aliohjelma

DiskError:
MsgBox (Err.Description), , "Latausvirhe"
Retries = Retries + 1     'suurenna  virhelaskurin  arvoa
If Retries >= 2 Then
    Resume Next
Else
    Resume
End If
```

Tämä on käyttökelpoinen menettely, jos kyseessä on virhe, jonka käyttäjä saattaa pystyä korjaamaan. On tärkeä muistaa, että Resume-lauseella *toistetaan* virheen aiheuttanut lause, kun taas Resume Next -lauseella *ohitetaan* virheen aiheuttanut lause ja jatketaan seuraavalta riviltä. Kun käytät Resume Next -lausetta, varmista, että seuraava lause on todella tarkoitus suorittaa. Kun ohjelman suoritus jatkuu, varmista myös, että virheenkäsittelijää ei vahingossa suoriteta uudelleen. Hyvä tapa ohittaa virheenkäsittelijä on käyttää Exit Sub -lausetta. Voit myös kirjoittaa Resume Next -lauseen, joka aiheuttaa siirtymisen virheenkäsittelijän jälkeiseen lauseeseen.

Jos haluat jatkaa seuraavaan harjoitukseen

➤ Jätä Visual Basic käyntiin ja siirry harjoitukseen 8.

Jos haluat lopettaa Visual Basicin

➤ Valitse File-valikosta komento Exit.

Jos näyttöön tulee Save-valintaikkuna, napsauta Yes.

Harjoituksen tiivistelmä

Tehtävä	Ohje	Painike
Lisätä ohjelmaan uusi lomake	Napsauta Lisää lomake -painiketta ja valitse Form. *tai* Valitse Project-valikosta komento Add Form ja napsauta Open.	
Ladata lomake muistiin	Käytä Load-lausetta. Esimerkki: `Load Form2`	
Näyttää ladattu lomake	Käytä Show-metodia. Esimerkki: `Form2.Show`	
Luoda moodinen lomake	Kirjoita Show-lauseeseen luku 1. Esimerkki: `Form2.Show 1`	
Piilottaa lomake	Käytä Hide-metodia. Esimerkki: `Form2.Hide`	
Poistaa lomake muistista	Käytä Unload-lausetta. Esimerkki: `Unload Form2`	
Vaihtaa objektin nimi	Muuta objektin Name-ominaisuutta Properties-ikkunassa.	
Luoda MDI-ylilomake	Valitse Project-valikosta komento Add MDI Form.	
Luoda MDI-alilomake	Valitse Project-valikosta komento Add Form ja aseta lomakkeen MDIChild-ominaisuudeksi True.	
Lähettää tekstirivi tulostimeen	Käytä Printer-objektia ja Print-metodia. Esimerkki: `Printer.Print "Meripoika"`	
Muuttaa tulostusasetuksia	Määritä Printer-objektin ominaisuudet suorituksen aikana.	
Lopettaa tulostustyö	Käytä EndDoc-metodia. Esimerkki: `Printer.EndDoc`	

Tehtävä	Ohje
Tulostaa koko lomake	Käytä PrintForm-metodia. Esimerkki: `Form2.PrintForm`
Havaita ajonaikaiset virheet ohjelmassa	Ota virheenkäsittely käyttöön lauseella `On Error GoTo viite` *Viite* on virheenkäsittelijän nimi.
Käsitellä ajonaikaiset virheet	Kirjoita virheenkäsittelyaliohjelma (jossa on yleensä If...Then- tai Select Case -lauseita) On Error -lauseella määritetyn viitteen jälkeen. Tavallisesti virheenkäsittelijä määrittää ominaisuuksia ja näyttää ilmoituksia MsgBox-funktiolla.
Jatkaa suoritusta virheen jälkeen	Käytä lausetta Resume, Resume Next tai Resume *viite*.
Lopettaa aliohjelma ennen End Sub -lausetta	Käytä Exit Sub -lausetta.

Kun tarvitset tietoja aiheesta	Valitse Visual Basicin Help-valikosta Microsoft Visual Basic Help Topics, valitse Index-välilehti ja...
Lomakkeiden käyttö	Kirjoita **forms**.
MDI-lomakkeiden käyttö	Kirjoita **MDI forms**.
Printer-objektin käyttö	Kirjoita **Printer object**.
Print-metodi	Kirjoita **Print method**.
PrintForm-metodi	Kirjoita **PrintForm**.
Err-objekti	Kirjoita **Err**.
Visual Basicin virhekoodit	Kirjoita **errors**.

Tietoja seuraavasta harjoituksesta

Seuraavassa harjoituksessa opit lisäämään käyttöliittymään kuvioita ja graafisia painikkeita. Siinä kuvataan myös veto- ja pudotustoimintoja, äänen käsittelyä ja yksinkertaisten animaatioiden luomista.

Harjoitus

8

Kuvioiden ja erikoistehosteiden lisääminen

Arvioitu aika
55 min.

Tässä harjoituksessa opit

- lisäämään lomakkeeseen kuvioita Viiva- ja Kuvio-työkaluilla
- tekemään graafisia komentopainikkeita Kuva-työkalulla
- lisäämään ohjelmiisi veto -ja pudotustoimintojen tuen
- muuttamaan hiiren osoittimen muotoa
- tekemään erikoistehosteita animaatioilla.

Useimmille ohjelmoijille kuvien ja erikoistehosteiden lisääminen sovellukseen on ohjelmoinnin jännittävin ja kiehtovin osa. Vaikuttavien ja hyödyllisten graafisten tehosteiden muodostaminen Visual Basicilla on onneksi sekä palkitsevaa että helppoa. Tässä harjoituksessa opit lisäämään ohjelmiisi kiinnostavia lisäominaisuuksia. Näytän, miten piirrät lomakkeeseen kuvioita, teet graafisia komentopainikkeita ja muutat hiiren osoittimen ulkonäköä. Kuvaan myös, miten lisäät ohjelmiisi veto- ja pudotustoimintojen tuen ja miten saat aikaan yksinkertaisen animaation käyttämällä kuva- ja ajastinobjekteja. Kun olet käynyt läpi tämän harjoituksen, osaat jo rakentaa näyttävän käyttöliittymän.

Kuvien lisääminen käyttämällä Viiva- ja Kuvio-työkaluja

Viiva- ja Kuvio-työkaluilla piirretään geometrisiä kuvioita.

Olet jo nähnyt, miten lomakkeeseen lisätään bittikarttoja, kuvakkeita ja Windows-metatiedostoja luomalla kuvakehys- ja kuvaobjekteja. Valmiin kuvan lisääminen Visual Basic -ohjelmaan on helppoa, ja olet harjoitellut sitä jo useita kertoja. Nyt opit piirtämään lomakkeeseen kuvioita kätevillä Viiva- ja Kuvio-työkaluilla. Niillä voit tehdä piirroksia, joissa on monenmuotoisia ja -kokoisia kuvioita sekä eri värejä. Näillä työkaluilla piirretyissä objekteissa on muutamia rajoituksia: käyttäjä ei voi valita niitä ohjelman suorituksen aikana, ja ne eivät voi olla muiden objektien päällä. Ne ovat silti tehokkaita, nopeita ja helppokäyttöisiä.

Viiva-työkalu

Viiva-työkalu

Viiva-työkalulla piirretään lomakkeeseen suora viiva. Sitten voit muuttaa viivan ulkoasua määrittämällä erilaisia ominaisuuksia samalla tavoin kuin muidenkin objektien ominaisuuksia. Viivaobjektin tärkeimmät ominaisuudet ovat BorderWidth, BorderStyle, BorderColor ja Visible.

BorderWidth-ominaisuus muuttaa viivan leveyttä. Se on hyödyllinen varsinkin, jos lisäät tekstin alle viivan tai erotat viivalla kaksi objektia toisistaan. BorderStyle-ominaisuus muuttaa viivan jatkuvaksi, piste- tai katkoviivaksi, ja BorderColor-ominaisuus valitsee viivan väriksi minkä tahansa Visual Basicin vakiovärin. Visible-ominaisuus puolestaan määrää, onko viiva näkyvä vai näkymätön.

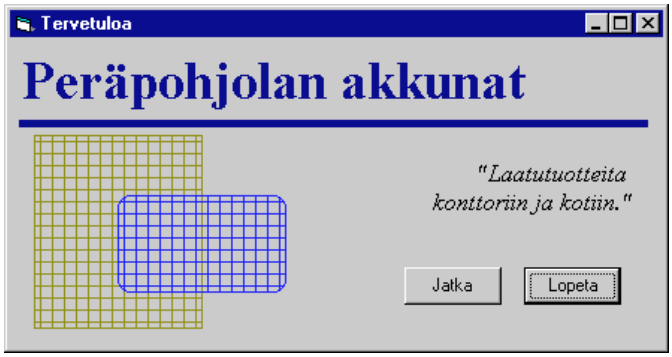
Kuvio-työkalu

Kuvio-työkalulla lisätään lomakkeisiin suorakulmioita, neliöitä, soikioita ja ympyröitä. Piirrä Kuvio-työkalulla haluamasi kuvio ja määritä sitten sen ominaisuudet Properties-ikkunassa. Shape-ominaisuus vaikuttaa piirretyn kuvion muotoon: voit valita suorakulmion, pyöristetyn suorakulmion, neliön, pyöristetyn neliön, soikion tai ympyrän. Voit muodostaa monimutkaisiakin kuvia piirtämällä useita viivoja ja kuvioita.

Kuvio-työkalu

Kuvio-objektin tärkeitä ominaisuuksia ovat FillColor, joka määrää objektin värin, FillStyle, joka määrää täytevärin kuvioinnin, ja BorderColor, joka määrää kuvion reunaviivan värin. Kuvio-objektilla on myös Visible-ominaisuus, jolla objekti tuodaan näkyviin tai piilotetaan.

Seuraavaksi pääset käytännössä kokeilemaan Viiva- ja Kuvio-työkaluja. Piirrä niillä aloitusikkuna ohjelmaan, jota käyttää kuvitteellinen yritys nimeltä Perä-pohjolan akkunat. Aloitusikkuna on esitetty seuraavalla sivulla.



Viiva- ja Kuvio-työkalujen käyttö

1 Käynnistä Visual Basic.

Jos Visual Basic on jo käynnissä, valitse File-valikosta komento New Project ja avaa uusi vakioprojekti.

2 Levennä lomaketta, jotta siinä on tilaa suurelle tekstille, kuvio-objekteille ja komentopainikkeille.

3 Napsauta Teksti-työkalua ja lisää lomakkeen yläosaan leveä tekstikehys.

Teksti-työkalu

4 Avaa Properties-ikkuna ja määritä tekstiobjektin Caption-ominaisuudeksi ”Peräpohjolan akkunat”.

5 Määritä tekstiobjektin Font-ominaisuudeksi 26 pisteen lihavoitu (Bold) Times New Roman. Valitse ForeColor-ominaisuudella edustaväriksi tummansininen (Dark blue).

Tekstiobjektin pitäisi olla lähes lomakkeen levyinen. Muuta sen leveyttä ja korkeutta tarvittaessa niin, että teksti mahtuu yhdelle riville.

6 Napsauta Viiva-työkalua ja piirrä yrityksen nimen alapuolelle viiva. Viivaobjektin pitäisi ulottua lähes lomakkeen reunasta toiseen samoin kuin tekstiobjekti.

Viiva-työkalu

Piirtämäsi viivan kummassakin päässä näkyy valintakahvat. Niillä voit tarvittaessa muuttaa viivan pituutta.

7 Napsauta Properties-ikkuna-painiketta, jotta Properties-ikkuna avautuu, ja määritä viivaobjektin ominaisuudet näin:

Properties-ikkuna-painike

Objekti	Ominaisuus	Asetus
Line1	BorderWidth	5
	BorderColor	Dark blue

Uuden viivaobjektin nimi on Line1. BorderWidth-asetus määrää viivan leveydeksi 5 twip-yksikköä (1 twip on 1/20 pistettä eli 1/1440 tuumaa eli noin 0,0176 mm.) BorderColor-asetus muuttaa viivan tummansiniseksi.

Piirrä lomakkeeseen kaksi ikkunaruudun kuvaa.

Kuvio-työkalu

8 Napsauta Kuvio-työkalua ja piirrä suorakulmio lomakkeen vasempaan reunaan.

Tämä suorakulmio on ensimmäisen ikkunaruudun reunaviiva. Määrität kohta FillStyle-ominaisuuden, joka muuttaa suorakulmion enemmän ikkunaruudun näköiseksi.

9 Napsauta Kuvio-työkalua uudelleen ja piirrä toinen suorakulmio niin, että se on osittain ensimmäisen suorakulmion päällä.

Kuvio-työkalu toimii suunnilleen samoin kuin grafiikkaohjelman yleiskäyttöinen piirtotyökalu. Sillä luodaan perusmuoto, ja tätä muokataan määrittämällä ominaisuuksia.

10 Napsauta Properties-ikkuna-painiketta ja määritä suorakulmioiden ominaisuudet näin:

Objekti	Ominaisuus	Asetus
Shape1	Shape	0 - Rectangle
	FillColor	Dark yellow
	FillStyle	6 - Cross
	BorderColor	Dark yellow
Shape2	Shape	4 - Rounded rectangle
	FillColor	Light blue
	FillStyle	6 - Cross
	BorderColor	Light blue

11 Lisää lomakkeen oikeaan reunaan tekstikehys. Tee siitä aika kapea, jotta siihen kirjoitettava teksti jakautuu kahdelle riville. Määritä sen ominaisuudet näin:

Objekti	Ominaisuus	Asetus
Label2	Caption	""Laatutuotteita konttoriin ja kotiin.""
	Font	Times New Roman, Italic, 12 pistettä
	Alignment	1 - Right Justify

Komentopainike-työkalu

12 Napsauta Komentopainike-työkalua ja lisää lomakkeen oikeaan alakulmaan komentopainike. Piirrä sen vasemmalle puolelle toinen komentopainike.

13 Määritä komentopainikkeiden ominaisuudet näin:

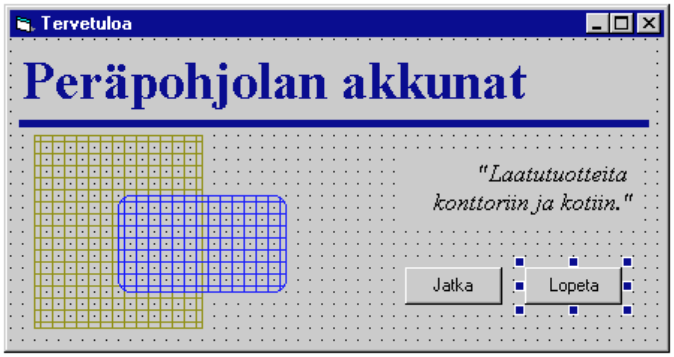
Objekti	Ominaisuus	Asetus
Command1	Caption	”Lopeta”
Command2	Caption	”Jatka”

14 Kaksoisnapsauta Lopeta-painiketta, kirjoita tapahtuma-aliohjelmaan Command1_Click lause **End** ja sulje Code-ikkuna.

Piirtämäsi aloituslomake on tarkoitettu toivottamaan tervetulleeksi ohjelmaan, mutta jos käyttäjä haluaa lopettaa heti, Quit-painike antaa tähän mahdollisuuden. Koska aloituslomake on toistaiseksi ohjelman ainoa osa, ainoa vaihtoehto on lopettaa ohjelma napsauttamalla Quit-painiketta.

15 Vaihda lomakkeen otsikoksi eli Caption-ominaisuudeksi ”Tervetuloa ja muuta lomakkeen ja objektien kokoa niin, että tulos näyttää sopusuhtaiselta.

Valmiin lomakkeen pitäisi näyttää suunnilleen tällaiselta:



Voit käyttää tällaista aloituslomaketta missä tahansa kirjoittamassasi ohjelmassa. Sen tehtävä on toivottaa käyttäjä tervetulleeksi ja antaa mahdollisuus jatkaa tai lopettaa. Kun käyttäjä napsauttaa Jatka-painiketta, ohjelma piilottaa aloituslomakkeen ja näyttää sovelluksen päälomakkeen.

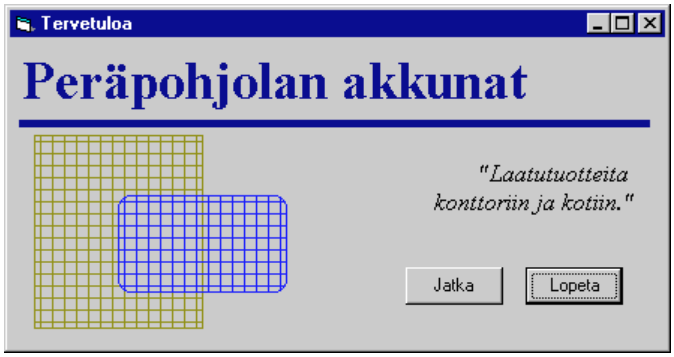
StartFrm-ohjelman suorittaminen

1 Napsauta työkalurivin Käynnistä-painiketta.

Aloituslomake tulee näkyviin seuraavan sivun kuvan kaltaisena.

Käynnistä-painike

Valmis StartFrm-ohjelma on kansiossa \Vb5Sbs\Less08.



Viiva- ja kuvio-objektit tuovat hieman ilmettä aloituslomakkeeseen. Ne tulevat aina näkyviin nopeammin kuin levyltä ladattavat kuvatiedostot.

2 Lopeta ohjelma napsauttamalla Lopeta-painiketta.

Tallenna projekti -painike

3 Napsauta työkalurivin Tallenna projekti -painiketta ja tallenna lomake Less08-kansioon nimellä **MyStartFrm.frm**. Tallenna projekti nimellä **MyStartFrm.vbp**.

Muotojen piirtäminen grafiikkametodeilla

Visual Basicissa on useita metodeja, joilla lisätään ohjelmiin piirroksia ja visuaalisia tehosteita. Grafiikkametodit ovat tapahtuma-aliohjelmissa käytettäviä lauseita, joilla piirretään lomakkeeseen kuvia tai lähetetään kuvia tulostimeen.

Grafiikkametodien haitta on se, että ne edellyttävät huomattavasti suunnittelu- ja ohjelmointityötä. Sinun pitää opetella lauseiden kielioppi, tuntea lomakkeessa käytettävä koordinaatisto ja päivittää kuvat, jos ne jäävät toisen ikkunan taakse. Toisaalta voit tehdä grafiikkametodeilla tehosteita, jotka eivät ole mahdollisia Viiva- tai Muoto-työkalulla, esimerkiksi kaaria tai yksitellen piirrettyjä pisteitä.

Hyödyllisimpiä grafiikkametodeja ovat Line, joka piirtää viivan tai tyhjän tai täytetyn suorakulmion, Circle, joka piirtää ympyrän, soikion tai ympyräsektorin, sekä PSet, joka määrittää näytön yhden pisteen värin.

Esimerkiksi seuraava Circle-lause piirtää lomakkeeseen ympyrän, jonka säde on 750 twip-yksikköä ja jonka (x, y)-koordinaatit ovat (1500, 1500):

```
Circle (1500, 1500), 750
```

Jos haluat tietää enemmän grafiikkametodeista, etsi Visual Basicin ohjeista hakusanaa *line, circle* tai *pset*.

Graafisten komentopainikkeiden luominen

Olet käyttänyt jo useita kertoja komentopainikkeita, jotka ovat intuitiivinen keino valita komentoja. Komentopainike tehdään piirtämällä se Komentopainike-työkalulla ja sitten kirjoittamalla sen Caption-ominaisuudeksi sana, joka kuvaa painikkeella käynnistettävää toimintoa.

Voit piirtää Kuva-työkalulla työkaluriviä muistuttavia komentopainikkeita.

Visual Basicissa on vaihtoehtona tekstipainikkeille Kuva-työkalu, jolla lisätään ohjelmiin graafisia komentopainikkeita. Graafinen komentopainike eli kuvake sisältää kuvan, joka esittää painikkeella käynnistettävää toimintoa. Esimerkiksi levykkeen kuvan sisältävä painike voisi edustaa komentoa, jolla tallennetaan tiedosto levylle. Graafiset komentopainikkeet voidaan sijoittaa ohjelmaan yksitellen tai koota ryhmiksi, joita sanotaan *työkaluriveiksi*. Visual Basicin työkalurivi on esimerkki tällaisesta painikeryhmästä. Seuraavassa tehtävässä opit piirtämään aitoja graafisia komentopainikkeita, jotka ”painuvat alas” ja ”ponnahtavat ylös”, kun napsautat niitä. Näin toimivat kaikkien Windows-sovellusten painikkeet.

MouseDown-tapahtuman havaitseminen

Jotta graafiset komentopainikkeet näyttäisivät ja tuntuisivat todellisilta, ohjelman pitää reagoida heti, kun käyttäjä siirtää osoittimen painikkeen päälle ja pitää hiiren painiketta alhaalla. Tähän mennessä olet käyttänyt hiiren toimintojen tunnistamiseen Click-tapahtumaa, mutta tässä tapauksessa se ei riitä. Ohjelman pitää reagoida painiketta *painettaessa*, ei painikkeen vapauttamisen jälkeen. Voit tarkkailla hiiren toimintoja Visual Basicissa käyttämällä MouseDown-tapahtumaa.

MouseDown-tapahtuma havaitsee kaksoisnapsautuksen ensimmäisen osan.

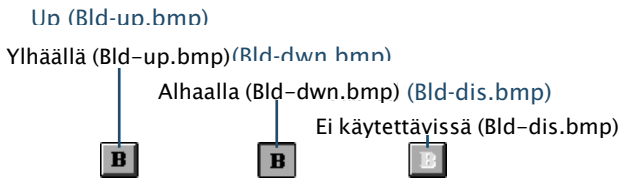
MouseDown on erityistapahtuma, joka käynnistää tapahtuma-aliohjelman, kun käyttäjä siirtää osoittimen lomakkeen objektin päälle ja pitää hiiren painiketta alhaalla. Jos kirjoitat MouseDown-tapahtumaa varten aliohjelman (esimerkiksi Image1_MouseDown), ohjelma osaa toimia oikealla tavalla, kun käyttäjä pitää hiiren painiketta alhaalla ja osoittaa objektia. Lisäksi ohjelman pitää muuttaa käyttäjän napsauttaman painikkeen ulkoasua, jotta painike näyttää painuvan alas.

HUOMAA Ohjelma voi tunnistaa MouseDown-tapahtumien lisäksi MouseUp-tapahtumia, jotka aiheutuvat hiiren painikkeen vapauttamisesta, ja MouseMove-tapahtumia, jotka aiheutuvat hiiren siirtämisestä.

Painikkeiden kuvien vaihtaminen

Miten komentopainike saadaan näyttämään kohoavalta tai painuvalta, kun sitä napsautetaan? Kuten voit arvata, komentopainikkeessa näkyvä kuva ei muutu käyttäjän napsauttaessa painiketta, vaan se korvataan toisella kuvalla MouseDown-

tapahtuma-aliohjelmassa. Seuraavasta kuvasta näkyy, että graafisella komentopainikkeella on kolme tilaa: ylhäällä, alhaalla ja ei käytettävissä.



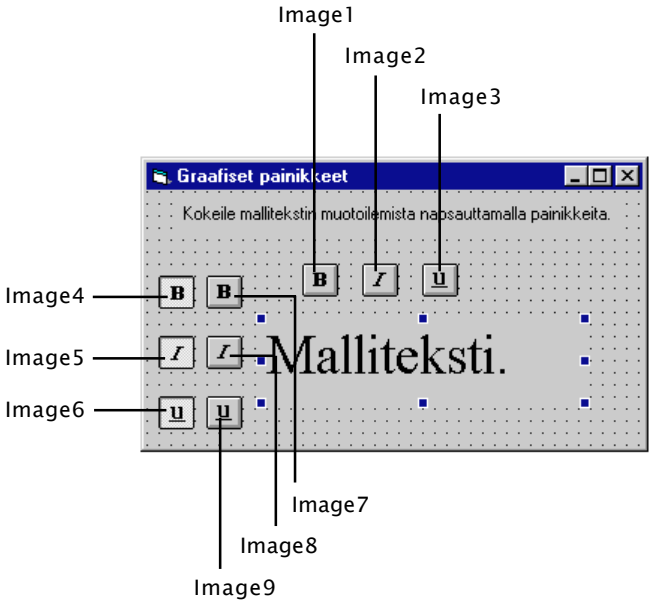
”Ylhäällä” on normaalitila, jossa painiketta ei ole painettu eli se näyttää olevan koholla alustastaan. ”Alhaalla” on tila, jossa painike on valittu eli aktiivinen ja painunut alas. ”Ei käytettävissä” on ylimääräinen tila, jota tarvitaan, kun painiketta ei jostakin syystä voi käyttää. Kaikissa komentopainikkeissa ei ole tätä tilaa.

Voit päivittää painikkeiden tilat käyttämällä MouseDown-aliohjelmaa.

Visual Basic -ohjelma muuttaa komentopainikkeiden tilaa vaihtamalla kuvaa kuvaobjektissa, johon painike on sijoitettu. Kuvaobjektin MouseDown-aliohjelma huolehtii kuvan vaihtamisesta. Jotta aliohjelma toimisi oikein, sen on luettava painikkeen nykyinen tila (ylhäällä, alhaalla tai ei käytettävissä), valittava uusi tila ja suoritettava pyydetty tehtävä (esimerkiksi tekstin lihavoiminen). Painikkeiden kuvat voidaan ladata muistiin suorituksen aikana LoadPicture-funktiolla tai vaihtaa lomakkeessa käyttämällä sijoituslauseita. Eräät ohjelmoijat sijoittavat kaikki kolme eritilaista painiketta lomakkeeseen, jotta päivitys toimii nopeammin.

HUOMAA Voit lisätä graafisia komentopainikkeita ja työkalurivejä mihin tahansa Visual Basic -lomakkeeseen, mutta MDI-lomakkeissa on erikoisominaisuuksia, jotka helpottavat painikeryhmien käsittelemistä. Lisäksi Visual Basicin Professional- ja Enterprise-versioissa on Toolbar-objekti, jolla on kätevä luoda ja hallita MDI-lomakkeiden työkalurivejä.

Seuraava tehtäväsi on kirjoittaa ohjelma, jossa on kolme graafista komento-painiketta (Lihavointi, Kursivointi ja Alleviivaus) lomakkeen tekstin muotoile-miseen. Ohjelma esittää komentopainikkeet käyttäen kuutta kuvatiedostoa, jotka ovat Less08-kansiossa, päivittää painikkeiden tilat ja muotoilee tekstin käyttäen kolmea MouseDown-aliohjelmaa. Ohjelmassa tarvittava lomake on esitetty seuraavassa kuvassa.



Tee painikkeet, joilla lihavoidaan, kursivoidaan tai alleviivataan teksti.

Teksti-työkalu

Kuva-työkalu

Graafisten komentopainikkeiden luominen

1. Valitse File-valikosta komento New Project ja luo vakiotiedosto napsauttamalla OK.
2. Muuta lomake tavanomaisen valintaikkunan kokoiseksi.
3. Napsauta Teksti-työkalua ja lisää lomakkeen yläreunaan pitkä tekstiobjekti.
4. Piirrä Kuva-työkalulla kolme pientä kuvaobjektia vierekkäin lomakkeen keskelle, tekstiobjektin alapuolelle.

   Näihin kuvaobjekteihin sijoitetaan ohjelmassa tarvittavat lihavointi-, kursivointi- ja alleviivauspainikkeet.
5. Lisää lomakkeen vasempaan reunaan kuusi kuvaobjektia. Katso niiden ehdotettu sijainti yllä olevasta kuvasta.

   Näihin kuvakehyksiin sijoitetaan jokaisesta painikkeesta kaksi eritilaista kuvaa, jotta ne voidaan sijoittaa kolmeen ensimmäiseen kuvaobjektiin. Toinen vaihtoehto olisi ladata painikkeiden kuvat levyltä tai tallentaa osa tiloista yleismuuttujiin, mutta tässä esimerkissä kaikkien kuvien sijoittaminen lomakkeeseen on nopea ja kätevä ratkaisu.
6. Piirrä lomakkeen keskelle suuri tekstikehys, johon kirjoitetaan malliteksti.

Tämä teksti on tarkoitus muotoilla napsauttamalla komentopainikkeita. Tekstin ulkoasua muutetaan määrittämällä tekstiobjektin FontBold-, FontItalic- ja FontUnderline-ominaisuudet MouseDown-aliohjelmissa.

Tag-ominaisuuteen voidaan sijoittaa objektin tunnistamista helpottava teksti.

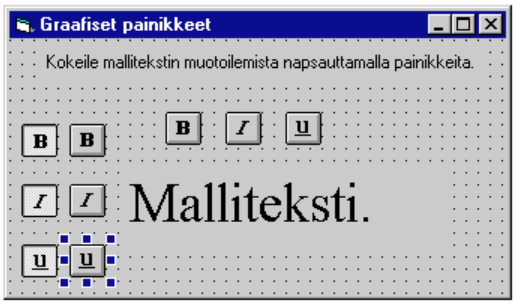
Määritä seuraavaksi lomakkeen objektien ominaisuudet. Muuta ensin kuusi jälkimmäistä kuvaobjektia näkymättömiksi ja muokkaa ja muotoile tekstiobjektien tekstit. Kirjoita sitten painikkeiden alkutilojen tunnukset kuvaobjektin erikoisominaisuuteen *Tag*. Tag on ominaisuus, johon voidaan kirjoittaa objektia kuvaavia huomautuksia. Siihen sijoitetaan usein objektin nimi, mutta tässä ohjelmassa siihen tallennetaan painikkeen tila eli "Up" tai "Down".

7 Määritä objektien ominaisuudet näin:

Objekti	Ominaisuus	Asetus
Label1	Caption	"Kokeile mallitekstin muotoilemista napsauttamalla painikkeita."
Label2	Caption Font	"Malliteksti." Times New Roman, 28 pistettä
Form1	Caption	"Graafiset painikkeet"
Image1	Picture Tag	"c:\vb5sbs\less08\bld-up.bmp" "Up"
Image2	Picture Tag	"c:\vb5sbs\less08\itl-up.bmp" "Up"
Image3	Picture Tag	"c:\vb5sbs\less08\ulin-up.bmp" "Up"
Image4	Picture Visible	"c:\vb5sbs\less08\bld-dwn.bmp" False
Image5	Picture Visible	"c:\vb5sbs\less08\itl-dwn.bmp" False
Image6	Picture Visible	"c:\vb5sbs\less08\ulin-dwn.bmp" False
Image7	Picture Visible	"c:\vb5sbs\less08\bld-up.bmp" False
Image8	Picture Visible	"c:\vb5sbs\less08\itl-up.bmp" False
Image9	Picture Visible	"c:\vb5sbs\less08\ulin-up.bmp" False

HUOMAA Kun määrität Picture-ominaisuuden, kuvaobjektin koko muuttuu levyltä luettavan kuvan mukaiseksi.

Valmiin lomakkeen pitäisi näyttää tällaiselta:



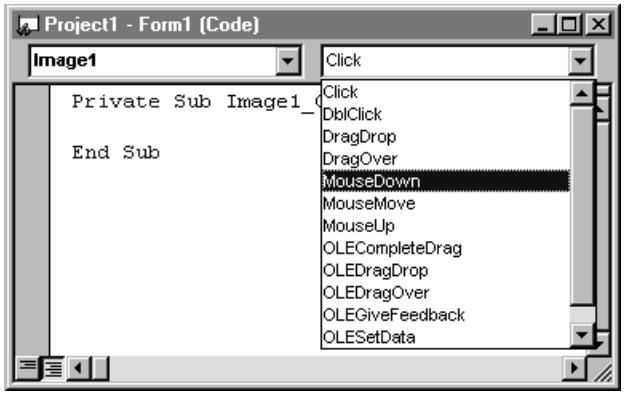
Kirjoita seuraavaksi kolmen MouseDown-aliohjelman lauseet.

8 Kaksoisnapsauta Image1-objektia eli mallitekstin yläpuolella olevaa lihavointipainiketta.

Tapahtuma-aliohjelma Image1_Click ilmestyy Code-ikkunaan. Tällä kertaa sinun pitää kirjoittaa lauseet MouseDown-tapahtumaa eikä Click-tapahtumaa varten. Kun haluat avata objektin jonkin toisen tapahtuma-aliohjelman, valitse tapahtuma avattavasta aliohjelmaluettelosta.

9 Avaa Code-ikkunan oikeassa yläkulmassa oleva aliohjelmaluettelo.

Luettelossa näkyvät tapahtumat, jotka kuvaobjekti pystyy tunnistamaan.



10 Valitse luettelosta MouseDown-tapahtuma.

MouseDown-aliohjelma sisältää neljä muuttujaa, joita voit käyttää ohjelmissasi.

Tapahtuma-aliohjelma Image1_MouseDown ilmestyy Code-ikkunaan. Se saa MouseDown-tapahtumasta neljä tietoa: hiiren painikkeen tunnuksen, alhaalla pidetyn ALT-, CTRL- ja VAIHTONÄPPÄIMEN yhdistelmän (ehkä), osoittimen vaaka- eli x-koordinaatin ja y-eli pystykoordinaatin. Nämä

arvot palautetaan eli *välitetään* tapahtuma-aliohjelmaan muuttujissa, joihin voit viitata ohjelmalauseissa. Vaikka et tarvitse tapahtuma-aliohjelman muuttujia eli *argumentteja* tässä ohjelmassa, niistä voi olla paljon hyötyä, ja monet tapahtuma-aliohjelmat käyttävät niitä. Opit hieman lisää parametreistä tämän harjoituksen myöhemmässä ohjelmassa DragDrop.

11 Kirjoita Image1_MouseDown-aliohjelmaan seuraavat lauseet:

FontBold-ominaisuus määrää, onko teksti lihavoitu.

```
If Image1.Tag = "Up" Then
     Image1.Picture = Image4.Picture
     Label2.FontBold = True
     Image1.Tag = "Down"
Else
     Image1.Picture = Image7.Picture
     Label2.FontBold = False
     Image1.Tag = "Up"
End If
```

Tällä yksinkertaisella If-rakenteella käsitellään kaksi erilaista lihavointipainikkeen napsautusta. Jos painike on ylhäällä, aliohjelma vaihtaa kuvan Bld-up.bmp tilalle kuvan Bld-dwn.bmp, muuttaa tekstin lihavoiduksi ja sijoittaa kuvaobjektin Tag-ominaisuuteen tekstin ”Down”. Jos painike on alhaalla, aliohjelma vaihtaa kuvan Bld-dwn.bmp tilalle kuvan Bld-up.bmp, poistaa lihavoinnin tekstistä ja sijoittaa Tag-ominaisuuteen tekstin ”Up”. Aliohjelma siis vaihtaa painikkeen tilan vastakkaiseksi.

12 Avaa Code-ikkunan objektiluettelo ja valitse Image2-objekti. Avaa aliohjelmaluettelo ja valitse MouseDown-tapahtuma.

Tapahtuma-aliohjelma Image2_MouseDown tulee näkyviin.

13 Kirjoita seuraavat ohjelmalauseet:

Tag-ominaisuudella saadaan selville painikkeen tila.

```
If Image2.Tag = "Up" Then
     Image2.Picture = Image5.Picture
     Label2.FontItalic = True
     Image2.Tag = "Down"
Else
     Image2.Picture = Image8.Picture
     Label2.FontItalic = False
     Image2.Tag = "Up"
End If
```

Nämä lauseet määrittävät kursivointipainikkeen toiminnan. Ne ovat melkein samanlaiset kuin Image1_MouseDown-aliohjelmassa. Ainoat erot ovat kuvaobjektien nimissä sekä siinä, että FontBold-ominaisuuden tilalla on FontItalic-ominaisuus.

14 Avaa objektiluettelo ja valitse Image3-objekti. Avaa aliohjelmaluettelo ja valitse MouseDown-tapahtuma. Kun tapahtuma-aliohjelma Image3_MouseDown tulee näkyviin, kirjoita seuraavat ohjelmalauseet:

Picture-ominaisuudella vaihdetaan painikkeen kuva.

```
If Image3.Tag = "Up" Then
     Image3.Picture = Image6.Picture
     Label2.FontUnderline = True
     Image3.Tag = "Down"
Else
     Image3.Picture = Image9.Picture
     Label2.FontUnderline = False
     Image3.Tag = "Up"
End If
```

Nämä lauseet määrittävät alleviivauspainikkeen toiminnan. Ne ovat lähes samanlaiset kuin kahdessa edellisessä aliohjelmassa. Ohjelma on nyt valmis, joten voit tallentaa sen.

Tallenna projekti -painike

15 Napsauta työkalurivin Tallenna projekti -painiketta. Valitse kansio \Vb5Sbs\Less08 folder ja tallenna lomake nimellä **MyButtons.frm** ja projekti nimellä **MyButtons.vbp**.

Kokeile nyt ohjelmaa.

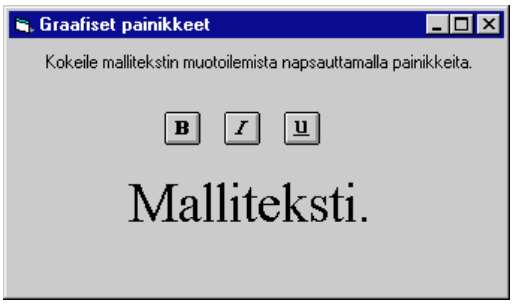
Ohjelman suorittaminen ja painikkeiden testaaminen

Käynnistä-painike

1 Napsauta työkalurivin Käynnistä-painiketta.

MyButtons-ohjelma näyttää tällaisen lomakkeen:

Valmis Buttons-ohjelma on kansiossa \Vb5Sbs\Less08.



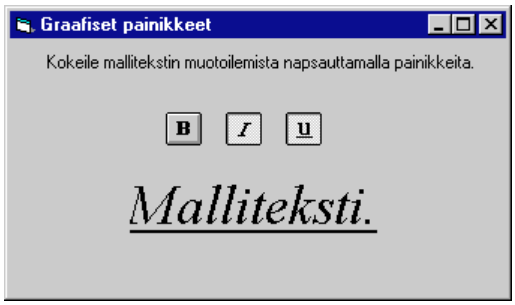
Lomakkeessa näkyy toimintaohje, komentopainikkeita ja malliteksti. Voit kokeilla kolmea painiketta miten monta kertaa ja missä järjestyksessä tahansa.

2 Napsauta kursivointipainiketta.

Kun painat hiiren painiketta, kursivointipainike ”painuu alas” ja teksti muuttuu kursivoiduksi.

3 Napsauta alleviivauspainiketta.

Lomakkeen pitäisi näyttää tällaiselta:



HUOMAA Jos haluat tekstin muotoilun muuttuvan vasta, kun hiiren painike vapautetaan, voit edelleen vaihtaa painikkeen kuvan MouseDown-aliohjelmassa, mutta vaihtaa fontin MouseUp-aliohjelmassa. Ohjelmoijana voit täydellisesti hallita, miten painikkeiden napsauttaminen vaikuttaa tekstiin.

4 Napsauta kursivointi- ja alleviivauspainikkeita uudelleen.

Painikkeet ”nousevat ylös”, ja teksti palautuu normaaliksi.

5 Kokeile myös lihavointipainikkeen toimintaa. Voit muotoilla tekstin käyttämällä painikkeiden yhdistelmiä.

6 Kun olet testannut painikkeet, napsauta työkalurivin Lopeta-painiketta.

Ohjelman suoritus päättyy, ja ohjelmointiympäristö palaa näkyviin.

Lopeta-painike

Vedon ja pudotuksen tuen lisääminen ohjelmiin

Vedon ja pudotuksen tuki voi tehdä käyttöliittymästä intuitiivisemman ja kätevämmän.

Windows-sovelluksissa käyttäjät valitsevat komentoja napsauttamalla valikoita ja painikkeita. Visual Basicilla voidaan lisätä ohjelmiin vaihtoehtoinen tapa suorittaa eräitä tehtäviä: *veto ja pudotus*. Tämä tarkoittaa sitä, että käyttäjä siirtää osoittimen objektin päälle, pitää hiiren painiketta alhaalla ja *vetää* objektin toiseen paikkaan. Sitten hän vapauttaa hiiren painikkeen, jolloin ohjelma siirtää objektin tai suorittaa muun tehtävän. Vetämällä ja pudottamalla voidaan esimerkiksi siirtää tekstiä paikasta toiseen tekstinkäsittelyohjelmassa tai poistaa näytöstä kohteita vetämällä ne ”roskakoriin”.

DragDrop-aliohjelma havaitsee objektin pudottamisen.

Veto- ja pudotustoimintoja ohjataan useilla ominaisuuksilla ja kahdella tapahtuma-aliohjelmalla. Määritä objektin DragMode-ominaisuudeksi 1, jotta käyttäjä voi vetää objektia. Voit myös määrätä DragIcon-ominaisuudella, että osoitin näyttää objektin vetämisen aikana samalta kuin objekti. Kun käyttäjä pudottaa objektin lomakkeeseen, Visual Basic vastaa tapahtumaan suorittamalla sen kohdeobjektin DragDrop-aliohjelman, johon objekti pudotettiin. Jos objekti vedetään lomakkeessa toisen objektin yli, Visual Basic suorittaa sen objektin DragOver-aliohjelman, jonka yli toista vedetään

Vetämisen ja pudottamisen vaiheet

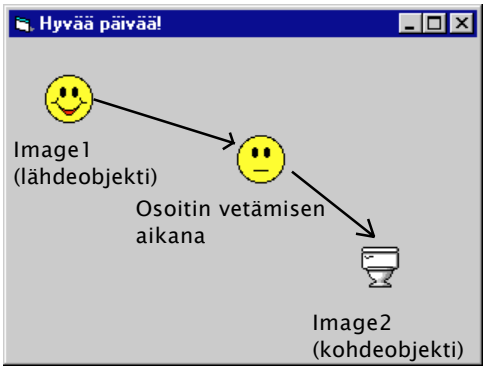
Kun lisäät ohjelmaan vedon ja pudotuksen tuen, suorita seuraavat tehtävät:

1. Salli objektin vetäminen ja pudottaminen. Visual Basic edellyttää, että otat vedon ja pudotuksen käyttöön objektikohtaisesti. Jos haluat, että objekti voidaan vetää ja pudottaa, määritä sen DragMode-ominaisuudeksi 1 ohjelmalauseella tai Properties-ikkunassa.

2. Valitse vetämisen aikana näytettävä osoitin. Visual Basic näyttää vedettävän objektin suorakulmiona, mutta voit korvata sen haluamallasi vetokuvakkeella. Määritä objektin DragIcon-ominaisuudeksi sopiva bittikartta tai kuvake ohjelmalauseella tai Properties-ikkunassa.

3. Kirjoita DragDrop- tai DragOver-aliohjelma kohdeobjektia varten.
Kirjoita tapahtuma-aliohjelma objektille, joka on vetämisen *kohde*. Visual Basic suorittaa sen objektin tapahtuma-aliohjelman, jonka yli objekti vedetään tai johon objekti pudotetaan. Tapahtuma-aliohjelman pitäisi suorittaa sopiva tehtävä, esimerkiksi muuttaa objektin paikkaa tai piilottaa objekti, antaa komento tai muuttaa kohdeobjektia jollakin tavalla. Avaa tapahtuma-aliohjelma napsauttamalla kohdeobjektia lomakkeessa, avaamalla Code-ikkuna ja valitsemalla sitten aliohjelmaluettelosta DragDrop- tai DragOver-tapahtuma.

Seuraava kuva havainnollistaa näitä ohjelmointivaiheita:



Form_Load

```
1. Image1.Picture = LoadPicture("c:\vb5sbs\less08\face03.ico")
2. Image1.DragMode = 1
3. Image1.DragIcon =
LoadPicture("c:\vb5sbs\less08\face01.ico")
```

Image2.DragDrop

```
4Image1.Visible = False
```

DragDrop-ohjelma

DragDrop-ohjelman polttotynnyri on paikka, johon käyttäjä voi heittää tarpeettomat esineet.

Seuraavasta ohjelmasta opit, miten ohjelmaan lisätään veto- ja pudotustoiminnot. Käyttäjä voi vetää lomakkeessa esitettyyn polttotynnyriin kolme esinettä ja sytyttää ne tuleen pudottamalla tynnyriin tulitikun. Polttotynnyri muistuttaa Microsoft Windows 95:n roskakoria tai Macintoshin roskatynnyriä. Voit lisätä ohjelmaasi polttotynnyrin, jotta käyttäjä voi hävittää erilaisia objekteja, kuten tarpeettomia asiakirjoja, tiedostoja, piirroksia, sähköpostiviestejä, verkkoyhteyksiä tai ikkunan osia. Ohjelma näyttää lomakkeessa kuvakehyksiä ja piilottaa objektit määrittämällä niiden Visible-ominaisuudeksi False.

Polttotynnyriohjelman luominen

1. Valitse File-valikosta komento New Project ja luo tavallinen suoritettava tiedosto napsauttamalla OK.
2. Muuta lomake tavanomaisen valintaikkunan kokoiseksi.
3. Napsauta Teksti-työkalua ja piirrä lomakkeen yläosaan pitkä tekstikehys. Tähän objektiin kirjoitetaan ohjelman käyttöohje.

Teksti-työkalu

4 Piirrä Kuva-työkalulla kuusi kuvaobjektia, jotka on esitetty seuraavassa kuvassa. Muista luoda objektit kuvan mukaisessa järjestyksessä. Näin varmistat sen, että seuraavalla sivulla esitetyt ominaisuudet sijoittavat kuhunkin kuvaobjektiin oikean kuvatiedoston.

Tag-ominaisuus "Fire" helpottaa tulitikkukuvakkeen tunnistamista.

5 Määritä ohjelman objektien ominaisuudet seuraavalla sivulla olevan taulukon mukaisesti. Huomaa Image3-objektin Tag-ominaisuuden arvo "Fire". Tämän tekstin perusteella ohjelma tunnistaa tulitikun, kun käyttäjä pudottaa sen tynnyriin.

HUOMAA Sinun täytyy määrittää Stretch-ominaisuus ennen Picture-ominaisuutta, jotta kuvakkeiden koot määräytyisivät oikein.

Objekti	Ominaisuus	Asetus
Label1	Caption	”Heitä kaikki tavarat pois ja pudota tulitikku.”
	Font	Times New Roman, Bold, 10 pist.
Form1	Caption	”Polttotynnyri”
Image1	Stretch	True
	Picture	”c:\vb5sbs\less08\trash02a.ico”
Image2	Picture	”c:\vb5sbs\less08\cdrom02.ico”
	DragIcon	”c:\vb5sbs\less08\cdrom02.ico”
	DragMode	1 - Automatic
Image3	Picture	”c:\vb5sbs\less08\fire.ico”
	DragIcon	”c:\vb5sbs\less08\fire.ico”
	DragMode	1 - Automatic
	Tag	”Fire”
Image4	Picture	”c:\vb5sbs\less08\gaspump.ico”
	DragIcon	”c:\vb5sbs\less08\gaspump.ico”
	DragMode	1 - Automatic
Image5	Picture	”c:\vb5sbs\less08\point11.ico”
	DragIcon	”c:\vb5sbs\less08\point11.ico”
	DragMode	1 - Automatic
Image6	Stretch	True
	Picture	”c:\vb5sbs\less08\trash02b.ico”
	Visible	False

Kun olet määrittänyt ominaisuudet, lomakkeen pitäisi näyttää tällaiselta:

6 Kaksoisnapsauta lomakkeen Image1-objektia (tyhjää polttotynnyriä).

Tapahtuma-aliohjelma Image1_Click ilmestyy Code-ikkunaan.

7 Avaa Code-ikkunan aliohjelmaluettelo ja valitse DragDop-tapahtuma.

DragDrop-aliohjelma palauttaa kolme parametriä, joita voit käyttää ohjelmassa: Source, X ja Y.

Tapahtuma-aliohjelma Image1_DragDrop tulee näkyviin. Sen Sub-lauseessa on mainittu kolme parametriä, jotka aliohjelma vastaanottaa objektin pudottamisen jälkeen: Source, X ja Y. Source-parametri ilmaisee vedetyn lähdeobjektin. Sen avulla lähdeobjekti piilotetaan, jotta vaikuttaa siltä kuin se olisi heitetty pois. Tässä aliohjelmassa ei tarvita parametrejä X ja Y.

8 Kirjoita tapahtuma-aliohjelmaan seuraavat ohjelmalauseet:

```
Source.Visible = False
If Source.Tag = "Fire" Then
    Image1.Picture = Image6.Picture
End If
```

Source-muuttuja ilmaisee tynnyriin pudotetun objektin.

Nämä ovat koko ohjelman ainoat lauseet. Ensimmäinen lause piilottaa vedetyn ja pudotetun objektin käyttämällä Source-muuttujaa ja Visible-ominaisuutta. Näin syntyy vaikutelma, että objekti on heitetty tynnyriin. Seuraavat lauseet tarkistavat, oliko tynnyriin heitetty objekti tulitikku-kuvake.

Muista, että ohjelman ominaisuuksia määrittäessäsi sijoitit Image3-objektin Tag-ominaisuuteen tekstin ”Fire” merkiksi siitä, että tämä objekti sisältää tulitikun kuvan. Aliohjelman If...Then-rakenne tarkistaa, oliko tynnyriin heitetty objekti tulitikku. Jos oli, aliohjelma ”sytyttää tulen” korvaamalla tyhjän tynnyrin kuvan palavan tynnyrin kuvalla.

9 Napsauta työkalurivin Tallenna projekti -painiketta. Valitse kansio \Vb5Sbs\Less08 folder ja tallenna lomake nimellä **MyDragDrop.frm**. Tallenna projekti nimellä **MyDragDrop.vbp**.

Tallenna projekti -painike

MyDragDrop-ohjelman suorittaminen

1 Napsauta työkalurivin Käynnistä-painiketta.

Käynnistä-painike

MyDragDrop-ohjelman ikkuna tulee näkyviin seuraavan sivun kuvan kaltaisena.

Valmis Drag-Drop-ohjelma on kansiossa \Vb5Sbs\Less08.

2 Vedä CD-levyaseman kuva polttotynnyriin ja vapauta hiiren painike.

Vetäessäsi hiiren osoitin muuttuu CD-levyaseman kuvan näköiseksi (DragIcon-ominaisuus toimii). Kun vapautat hiiren painikkeen polttotynnyrin yläpuolella, osoitin palautuu normaaliksi ja CD-levyaseman kuva katoaa.

Vetokuvake tulee näkyviin objektia vedettäessä.

3 Vedä ja pudota bensiinipumpun ja banaanin kuva polttotynnyriin.

Hiiren osoitin vaihtuu sen mukaan, kumpaa objektia vedät. (Bensiinipumppu saa varmasti liekit roihahtamaan.)

4 Pudota vielä tulitukku polttotynnyriin.

Heti vapautettuasi hiiren painikkeen polttotynnyri syttyy tuleen, kuten seuraavasta kuvasta näkyy:

VIHJE Tällä yleisellä veto- ja pudotusmenetelmällä on useita sovelluksia. Käytä sitä halutessasi osoittaa, että ohjelma poistaa objektia tai suorittaa jotakin muuta tehtävää. Voisit käyttää vetoa ja pudotusta, kun käsittelet kuvia, tulostat tiedostoja, lähetät fakseja ja sähköpostiviestejä, muokkaat tietokantoja tai järjestät verkon resursseja.

5 Lopeta ohjelma napsauttamalla otsikkorivin sulkemispainiketta.

Hiiren osoittimen vaihtaminen

DragDrop-ohjelmassa näit, miten DragIcon-ominaisuudella vaihdetaan hiiren osoitin vetämisen ja pudottamisen ajaksi. Voit myös valita yhden 12 valmiista osoittimesta MousePointer-ominaisuudella tai ladata oman osoittimen MouseIcon-ominaisuudella.

Valmiit osoittimet antavat käyttäjälle käsityksen siitä, mitä hänen pitäisi tehdä. Jos määrität MousePointer-ominaisuuden lomakkeen objektissa, osoitin muuttuu määritettyyn muotoon, kun käyttäjä siirtää sen kyseisen objektin päälle. Jos määrität MousePointer-ominaisuuden lomakkeessa, osoitin on koko ajan määrittämäsi muotoinen, ellei se ole sellaisen objektin päällä, jossa myös on määritetty MousePointer-ominaisuus.

Seuraavassa taulukossa on lueteltu muutamia osoittimen muotoja, joita voit valita MousePointer-ominaisuudella. (Löydät täydellisen luettelon Properties-ikkunasta.) Jos valitset muodon 99 (Custom), Visual Basic määrittää osoittimen muodon MouseIcon-ominaisuuden perusteella.

Osoitin	MousePointer-asetus	Kuvaus
+	2	Ristiosoitin piirtämistä varten
I	3	Tekstiosoitin tekstipohjaisiin sovelluksiin
✥	5	Koonmuutto-osoitin (valittavissa on osoittimia, joiden nuolet ovat erisuuntaiset)
⧗	11	Tiimalasiosoitin, joka kehottaa käyttäjää odottamaan
⊘	12	Pudotuskielto-osoitin, joka ilmaisee, että käyttäjän yrittämää tehtävää ei voi suorittaa

Animaation lisääminen ohjelmaan

Animaatio herättää ohjelman objektit "eloon".

Muuttuvat painikkeet ja objektien vetäminen lisäävät ohjelman visuaalista kiinnostavuutta, mutta ohjelmoijien mielestä graafisten tehosteiden kuningas on aina ollut animaatio. *Animaatio* tarkoittaa liikkeen jäljittelemistä näyttämällä nopeasti peräkkäin joukko samankaltaisia kuvia. Vetäminen ja pudottaminen on tavallaan "köyhän animaatio", koska sillä voidaan siirtää kuvia lomakkeessa paikasta toiseen. Todellinen animaatio sen sijaan edellyttää objektien siirtämistä ohjelmallisesti ja usein myös objektien koon tai muodon muuttamista.

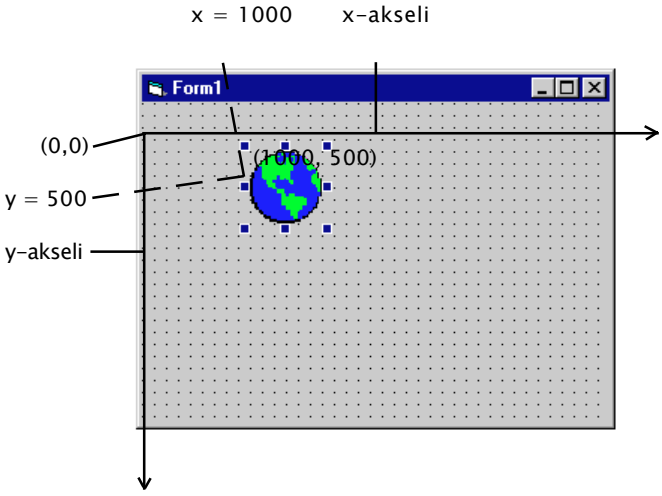
Seuraavaksi kerron, miten lisäät ohjelmaasi yksinkertaisen animaation. Näytän, miten Move-metodia käytetään, miten kuvakehyksen Top- ja Left-ominaisuudet päivitetään ja miten animaation nopeutta säädellään ajastinobjektilla.

Lomakkeen koordinaatisto

Animaatio-ohjelmalle on tyypillistä, että se siirtelee kuvia suhteessa näytön valmiiksi määritettyyn koordinaatistoon. Visual Basicin jokaisella lomakkeella on oma koordinaatistonsa. Sen alkupiste eli *origo* on lomakkeen vasemmassa yläkulmassa. Oletuskoordinaatisto koostuu riveistä ja sarakkeista, joiden mitta-yksikkö on laiteriippumaton twip (1/20 pistettä eli 1/1440 tuumaa eli noin 0,0176 mm).

Visual Basicin koordinaatisto perustuu lomakkeen riveihin ja sarakkeisiin.

Koordinaatiston rivit ovat x- eli vaaka-akselin suuntaiset ja sarakkeet y- eli pystyakselin suuntaiset. Koordinaatiston paikat määritetään ilmoittamalla rivin ja sarakkeen leikkauspiste merkinnällä (*x*, *y*). Vaikka mittayksiköksi voidaan vaihtaa jokin muu kuin twip, lomakkeen vasemman yläkulman koordinaatit ovat aina (0, 0). Seuraava kuva esittää, miten objektin sijainti kuvataan Visual Basicin koordinaatistossa:



Objektien siirtäminen koordinaatistossa

Move-metodilla siirretään objekteja.

Visual Basicissa on Move-niminen erikoismetodi, jolla siirretään objekteja koordinaatistossa. Sen perusmuoto on tällainen:

```
objekti.Move vasen, ylä
```

Objekti on siirrettävän objektin nimi, ja *vasen* ja *ylä* ovat objektin uuden paikan näyttökoordinaatit twip-yksiköinä. *Vasen* tarkoittaa objektin etäisyyttä lomakkeen vasemmasta reunasta ja *ylä* etäisyyttä yläreunasta. Move-metodilla voidaan muuttaa myös objektin korkeutta ja leveyttä. Esimerkki on tämän luvun kohdassa Lisätietoja.

Visual Basic -lause

```
Picture1.Move 1440, 1440
```

siirtää Picture1-objektin paikkaan (1440, 1440) näytössä eli tarkalleen 1 tuuman päähän lomakkeen vasemmasta reunasta ja yläreunasta.

Suhteelliset siirtymät määritetään objektin Left- ja Top-ominaisuuksilla.

Voit määrittää Move-metodilla myös suhteellisen siirtymän. Se tarkoittaa etäisyyttä, jonka verran objektin pitää siirtyä *nykyisestä paikastaan*. Kun on kyse suhteellisista koordinaateista, käytä objektin Left- ja Top-ominaisuuksia (jotka ilmaisevat objektin x- ja y-koordinaatit) sekä plus-operaattoria (+) ja miinus-operaattoria (−) Esimerkiksi lause

```
Picture1.Move Picture1.Left - 50, Picture1.Top - 75
```

siirtää Picture1-objektin nykyisestä paikastaan 50 twip-yksikköä lähemmäs vasenta reunaa ja 75 twip-yksikköä lähemmäs yläreunaa.

HUOMAA Move-metodin yhteydessä käytetään yleensä kuvakehysobjektia, koska se aiheuttaa vähemmän näytön välkkymistä kuin kuvaobjekti.

Animaation muodostaminen käyttämällä Move-metodia ja ajastinobjektia

Ajastinobjekti määrää liikkeen nopeuden.

Animaatio saadaan aikaan sijoittamalla yksi tai useita Move-metodeja ajastimen tapahtuma-aliohjelmaan siten, että ajastin aiheuttaa yhden tai usean objektin siirtymisen näytössä määritetyin aikavälein. Harjoituksessa 6 opit päivittämään kellonajan joka sekunti niin, että kello näytti ajan sekunnin tarkkuudella. Animaatiota tehtäessä ajastimen Interval-ominaisuuden pitää olla paljon pienempi; 1/5 sekuntia (200 millisekuntia), 1/10 sekuntia (100 millisekuntia) tai vähemmän. Tarkka aikaväli riippuu siitä, kuinka nopea animaation tulee olla.

Toinen tärkeä seikka on selvittää lomakkeen vasemman ja yläreunan sijainti Left- ja Top-ominaisuuksilla. Näiden arvojen perusteella tapahtuma-aliohjelma pystyy pysäyttämään animaation (poistamaan ajastimen käytöstä), kun objekti saavuttaa lomakkeen reunan. Voit myös käyttää Left- ja Top-ominaisuuksia If...Then- tai Select Case -rakenteessa niin, että objekti näyttää kimpoavan takaisin lomakkeen reunasta.

Savupilven lisääminen DragDrop-ohjelmaan

Voit luoda animaation Move-metodilla.

Seuraavassa ohjelmassa opit tekemään animaation käyttämällä kuvakehystä, Move-metodia ja ajastinobjektia. Tarkoitus on lisätä DragDrop-ohjelmaan savupilvi, joka tulee näkyviin, kun käyttäjä pudottaa tulitikun polttotynnyriin. Move-metodin ja ajastinobjektin ansiosta savupilvi näyttää ajelehtivan hiljalleen tuulessa, kunnes se häipyy lomakkeen ulkopuolelle.

Savuanimaation luominen

1. Valitse File-valikosta komento Save MyDragDrop.frm As ja tallenna DragDrop-lomake nimellä **MySmoke.frm**.
2. Valitse File-valikosta komento Save Project As ja tallenna DragDrop-projekti nimellä **MySmoke.vbp**.

   Kun tallennat lomakkeen ja projektin uusiin tiedostoihin, alkuperäinen DragDrop-ohjelma säilyy ennallaan.

Kuvakehys-työkalu

3. Napsauta Kuvakehys-työkalua ja piirrä pieni suorakulmio lomakkeen tyhjän polttotynnyrin yläpuolelle.

   Sijoitat savupilven kuvan tähän suorakulmioon ominaisuuksia määrittäessäsi.

Ajastin-työkalu

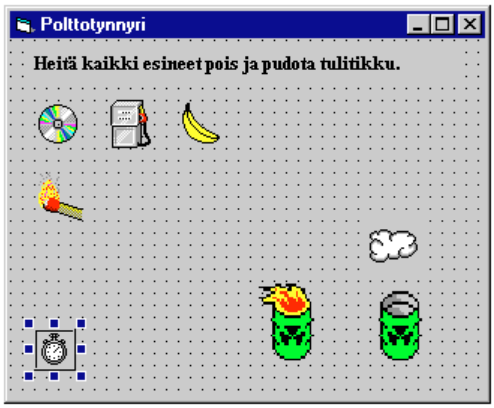
4. Napsauta Ajastin-työkalua ja lisää lomakkeen vasempaan alakulmaan ajastinobjekti.

   Ajastinobjektin (Timer1) koko muuttuu automaattisesti.

5. Määritä kuvakehyksen ja ajastimen ominaisuudet näin:

Objekti	Ominaisuus	Asetus
Picture1	Appearance	0 - Flat
	BackColor	Light gray
	BorderStyle	0 - None
	Picture	”c:\vb5sbs\less08\cloud.ico”
	Visible	False
Timer1	Enabled	False
	Interval	65

Kun olet määrittänyt ominaisuudet, lomakkeen pitäisi näyttää tällaiselta:



6 Kaksoisnapsauta tyhjän tynnyrin kuvaa eli Image1-objektia.

Tapahtuma-aliohjelma Image1_DragDrop ilmestyy Code-ikkunaan.

7 Muokkaa aliohjelma seuraavan listauksen mukaiseksi. (Neljäs ja viides rivi ovat uusia.)

```
Source.Visible = False
If Source.Tag = "Fire" Then
     Image1.Picture = Image6.Picture
     Picture1.Visible = True
     Timer1.Enabled = True
End If
```

Uudet lauseet muuttavat savupilven kuvan näkyväksi, kun polttotynnyri sytytetään, ja käynnistävät ajastimen, jotta pilvi alkaa liikkua. Koska olet jo määrittänyt ajastimen väliksi 65 millisekuntia, ajastin on käyttövalmis. Sinun pitää vain lisätä Move-metodi.

8 Avaa Code-ikkunan objektiluettelo ja valitse Timer1-objekti.

Tapahtuma-aliohjelma Timer1_Timer ilmestyy Code-ikkunaan.

9 Kirjoita seuraavat ohjelmalauseet:

```
If Picture1.Top > 0 Then
     Picture1.Move Picture1.Left - 50, Picture1.Top - 75
Else
     Picture1.Visible = False
     Timer1.Enabled = False
End If
```

Jos haluat pilven liukuvan oikealle tai alas, käytä Move-metodissa plusoperaattoria.

Kun ajastin on käytössä, tämä If...Then-rakenne suoritetaan yhä uudelleen 65 millisekunnin välein. Ensimmäinen lause tarkistaa, onko savupilvi saavuttanut lomakkeen yläreunan. Ellei ole (Top-ominaisuus on vielä positiivinen), aliohjelma käyttää suhteellista Move-metodia ja siirtää savupilveä 50 twip-yksikköä vasemmalle ja 75 twip-yksikköä ylöspäin.

Kuten voit huomata ohjelmaa suorittaesasi, pilvi vaikuttaa liukuvan hitaasti tuulen mukana. Jos haluat pilven siirtyvän oikealle tai alas, lisää Left- tai Top-ominaisuuteen positiivinen arvo. Kun pilvi on saavuttanut lomakkeen yläreunan, aliohjelman Else-osa muuttaa pilven näkymättömäksi ja poistaa ajastimen käytöstä eli lopettaa animaation.

10 Sulje Code-ikkuna ja tallenna muutokset napsauttamalla Tallenna projekti -painiketta.

Tallenna projekti -painike

11 Käynnistä ohjelma napsauttamalla työkalurivin Käynnistä-painiketta.

MySmoke-ohjelma käynistyy ohjelmointiympäristön alaisuudessa.

12 Vedä ja pudota polttotynnyriin CD-levyn, bensiinipumpun, banaanin ja lopuksi tulitikun kuva.

Polttotynnyri leimahtaa liekkeihin, ja savupilvi alkaa liikkua, kuten seuraavasta kuvasta näkyy:

Valmis Smoke-ohjelma on kansiossa \Vb5Sbs\Less08.

Hetken kuluttua pilvi on lipunut lomakkeen reunaan asti, ja animaatio pysähtyy.

13 Lopeta ohjelma napsauttamalla otsikkorivin sulkemispainiketta.

Nyt olet jälleen laajentanut ohjelmointitaitojasi — osaat lisätä ohjelmaan yksinkertaisen animaation. Kokeile Visual Basicin muitakin grafiikkatoimintoja omin päin, sillä se on paras tapa oppia ohjelmointia.

HUOMAA Jos haluat tutustua toiseen esimerkkiin animaatiosta, joka perustuu Move-metodiin, lataa muistiin ja käynnistä kansioon \Vb5Sbs\Less01 tallennettu StepUp-ohjelma. Se näyttää dynamiitti-pötkön, joka kimpoaa maasta muutaman kerran ja räjähtää savupilveksi. (Animaatiota ohjaavat lauseet ovat ajastimen tapahtuma-aliohjelmissa.) Muistat varmaan kokeilleesi StepUp-ohjelmaa ensimmäisessä harjoituksessa. Olet edennyt pitkälle noista ajoista!

Objektien suurentaminen ja pienentäminen ohjelman suorituksen aikana

Height- ja Width-ominaisuuksilla suurennetaan ja pienennetään objektia.

Haluatko kokeilla vielä yhtä erikoistehostetta? Visual Basic ylläpitää lomakkeen useimmista objekteista Top- ja Left-ominaisuuksien lisäksi Height- ja Width-ominaisuuksia. Näillä ominaisuuksilla voit näppärästi suurentaa ja pienentää objekteja ohjelmallisesti. Seuraavaksi näytän, miten sinun tulee menetellä.

Kuvakehyksen suurentaminen ohjelmallisesti

1. Valitse File-valikosta komento New Project ja luo vakiosovellus napsauttamalla OK.
2. Napsauta Kuva-työkalua ja piirrä lomakkeen vasempaan yläkulmaan pieni kuvakehys.

Kuva-työkalu

3. Määritä kuvakehyksen ja lomakkeen ominaisuudet seuraavan taulukon mukaisesti. Kun asetat kuvakehyksen ominaisuudet, huomaa Height- ja Width-ominaisuuksien nykyiset arvot. (Voit määrittää ne myös ohjelmaa kirjoittaessasi.)

Objekti	Ominaisuus	Asetus
Image1	Stretch	True
	Picture	"c:\vb5sbs\less08\earth.ico
Form1	Caption	"Lähestymme maata"

4. Kaksoisnapsauta Image1-objektia.

   Tapahtuma-aliohjelma Image1_Click ilmestyy Code-ikkunaan.

5. Kirjoita aliohjelmaan kaksi seuraavaa lausetta:

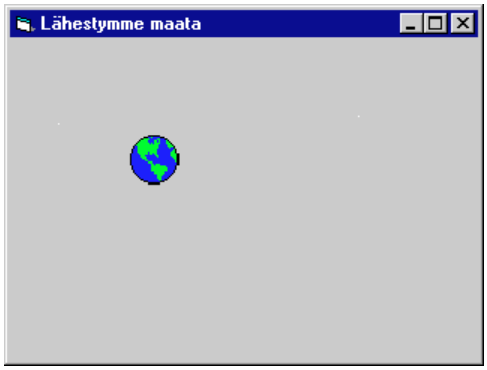
```
Image1.Height = Image1.Height + 200
Image1.Width = Image1.Width + 200
```

   Nämä lauseet suurentavat maapallon kuvan korkeutta ja leveyttä 200 twip-yksikön verran joka kerta, kun käyttäjä napsauttaa kuvakehystä. Jos käytät vähän mielikuvitusta, näyttää siltä kuin lähestyisit maapalloa avaruusaluksella.

Height- ja Width-ominaisuuksien suurentaminen kasvattaa maapallon kuvaa lomakkeessa.

6 Sulje Code-ikkuna ja käynnistä ohjelma napsauttamalla Käynnistä-painiketta.

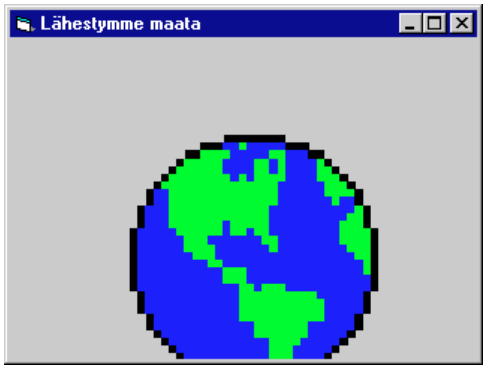
Lomakkeessa näkyy pieni maapallon kuva:



7 Napsauta maapallon kuvaa useita kertoja, jotta se suurenee.

Suunnilleen 10 napsautuksen jälkeen lomake näyttää tällaiselta:

Valmis Zoom-ohjelma on kansiossa \Vb5Sbs\Less08.



8 Kun avaruusalus tuntuu olevan sopivalla etäisyydellä maasta, lopeta ohjelma napsauttamalla sen otsikkorivin sulkemispainiketta.

Ohjelman suoritus päättyy, ja ohjelmointiympäristö palaa näkyviin.

9 Napsauta Tallenna projekti -painiketta ja tallenna lomake nimellä **MyZoom.frm** ja projekti nimellä **MyZoom.vbp**.

Lisätietoja: Ohjelman objektien nimeäminen

Nimet helpottavat käyttöliittymän osien tunnistamista.

Aiemmin tässä harjoituksessa teit MyButtons-ohjelman, joka havainnollisti graafisten komentopainikkeiden luomista ja käsittelemistä Visual Basic -ohjelmassa. Ohjelmassa oli yhdeksän kuvakehysobjektia (Image1–Image9) ja näiden näyttämistä ja ohjaamista varten kolme tapahtuma-aliohjelmaa. Ohjelmasta selviää paitsi se, miten graafisia painikkeita käsitellään, myös se, että objektien oletusnimet eivät ole kovin suositeltavia, kun ohjelmassa on monta samantyyppistä objektia. Jos sinä (tai joku muu ohjelmoija) tutkit ohjelmaa muutaman viikon kuluttua, joudut varmaan jonkin aikaa miettimään, mikä objekti tekee mitäkin.

Voit antaa objekteille selkeät ja helposti muistettavat nimet Name-ominaisuudella. Nimen täytyy alkaa kirjaimella, ja siinä saa olla enintään 40 merkkiä.

Samankaltaisista nimistä johtuvat sekaannukset vältetään antamalla jokaiselle objektille kuvaava nimi Name-ominaisuudella. Kunkin objektin nimen (samoin kuin muuttujan nimen) tulee selvästi ilmaista objektin käyttötarkoitus ja tyyppi. Yksiselitteiset ja intuitiiviset objektien nimet auttavat tunnistamaan objektit lomakkeessa ja ohjelmassa. Koska objektien nimiä käytetään tapahtuma-aliohjelmien nimissä ja ominaisuusasetuksissa, sinun pitäisi määrittää objektin Name-ominaisuus heti luotuasi objektin.

Objektien nimeämissääntöjä

Visual Basic -ohjelmoijilla on tapana helpottaa objektien tunnistamista kolmikirjaimisilla etuliitteillä.

Visual Basic -ohjelmoijien kesken on sovittu tietyistä kolmikirjaimisista etuliitteistä, joita tulisi käyttää kaikkien objektien nimissä. Näistä vakioetuliitteistä ilmenee objektin tyyppi. Kun noudatat seuraavia nimeämissääntöjä, muiden Visual Basic -ohjelmoijien on helpompi lukea kirjoittamiasi ohjelmia. Pystyt myös helpommin ymmärtämään Visual Basicin mukana toimitettuja mallisovelmia. Kolmas etu on se, että objektit näkyvät ryhmittäin aakkosjärjestyksessä Code-ikkunan objektiluettelossa.

Seuraavassa taulukossa on lueteltu objektien nimeämissääntöjä ja esimerkkejä. Voit harjoitella sääntöjen soveltamista seuraavassa tehtävässä.

Objekti	Etuliite	Esimerkki
avattava luetteloruutu	cbo	cboEnglish
valintaruutu	chk	chkReadOnly
komentopainike	cmd	cmdCancel
yleinen valintaikkuna	dlg	dlgOpen
tieto	dat	datBiblio
tietoihin sidottu avattava luetteloruutu	dbc	dbcEnglish
tietoihin sidottu luetteloruutu	dbl	dblPolicyCode
hakemistoluetteloruutu	dir	dirSource
levyasemaluetteloruutu	drv	drvTarget
tiedostoluetteloruutu	fil	filSource
kehys	fra	fraLanguage
lomake	frm	frmPrintForm
vaakavierityspalkki	hsb	hsbVolume
kuva	img	imgEmptyBarrel
teksti	lbl	lblInstructions
viiva	lin	linUnderline
luetteloruutu	lst	lstPeripherals
valikko	mnu	mnuFileOpen
OLE	ole	oleObject1
valintanappi	opt	optFrench
kuvakehys	pic	picSmokeCloud
kuvio	shp	shpWireScreen
tekstiruutu	txt	txtGetName
ajastin	tmr	tmrRunAnimation
pystyvierityspalkki	vsb	vsbTemperature

VIHJE Ohjelmoijat, jotka käyttävät Visual Basic for Applications, ohjelmointikieltä joka sisältyy Microsoft Wordiin, Exceliin, Accessiin, PowerPointiin ja Projectiin, valitsevat usein nimiä, joista ilmenee muuttujien tyypit ja Microsoft Office -sovelluksesta luettavat sisäiset vakiot.

Objektien nimien vaihtaminen Name-ominaisuudella

Teksti-työkalu

Properties-ikkuna-painike

1. Valitse File-valikosta komento New Project ja napsauta OK.
2. Piirrä Teksti-työkalulla kaksi tekstiobjektia lomakkeen keskialueelle, toinen lähelle yläreunaa ja toinen keskelle.
3. Napsauta työkalurivin Properties-ikkuna-painiketta ja määritä tekstiobjektien ja lomakeobjektin ominaisuudet seuraavan taulukon mukaisesti. Noudata annettuja nimeämisohjeita.

Objekti	Ominaisuus	Asetus
Label1	Caption	”Tervetuloa ohjelmaan!”
	Name	lblWelcome
Label2	Caption	”Poistu ohjelmasta napsauttamalla Lopeta.”
	Name	lblInstructions
Form1	Caption	”Nimeämissäännöt”
	Name	frmMainForm

Kun määrität Name-ominaisuudet, objektien nimet vaihtuvat sekä Properties-ikkunassa että ohjelman sisällä. Etuliite *lbl* tai *frm* ilmaisee, että kyseessä on teksti tai lomake, ja objektin nimen loppuosa kuvaa objektin merkitystä ohjelmassa.

Komentopainike-työkalu

4. Napsauta Komentopainike-työkalua ja piirrä lblInstructions-objektin alapuolelle komentopainike
5. Napsauta työkalurivin Properties-ikkuna-painiketta ja määritä komentopainikkeen ominaisuudet näin:

Objekti	Ominaisuus	Asetus
Command1	Caption	”Lopeta”
	Name	cmdQuit

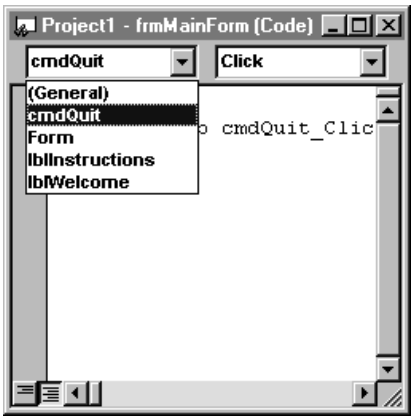
Etuliite *cmd* ilmaisee, että kyseessä on komentopainike, ja sana *Quit* kuvaa painikkeen toimintaa.

6. Kaksoisnapsauta cmdQuit-painiketta, jotta objektin tapahtuma-aliohjelma tulee näkyviin

Tapahtuma-aliohjelma cmdQuit_Click ilmestyy Code-ikkunaan. Visual Basic käyttää antamaasi nimeä objektin ”virallisena” nimenä.

7 Avaa Code-ikkunan objektiluettelo.

Luettelossa näkyy joukko objektien nimiä.



Objektien nimeämisestä saatava etu on ilmeinen. Sinä ja muut ohjelmoijat näkevät heti, mihin kukin objekti on tarkoitettu.

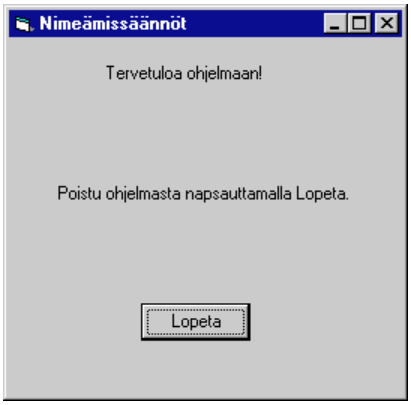
8 Sulje objektiluettelo painamalla ESC ja kirjoita cmdQuit_Click-aliohjelmaan **End**.

Käynnistä-painike

9 Sulje Code-ikkuna ja käynnistä ohjelma napsauttamalla työkalurivin Käynnistä-painiketta.

Lomakkeen pitäisi näyttää tällaiselta:

Valmis Name-Conv-ohjelma on kansiossa \Vb5Sbs\Less08.



10 Lopeta ohjelma napsauttamalla Lopeta-painiketta.

Ohjelman suoritus päättyy, ja ohjelmointiympäristö palaa näkyviin.

Tallenna projekti –painike

11 Napsauta työkalurivin Tallenna projekti -painiketta ja tallenna lomake kansioon \Vb5Sbs\Less08 nimellä **MyNameConv.frm** ja projekti nimellä **MyNameConv.vbp**.

TÄRKEÄÄ Helposti ymmärrettävien nimien antaminen objekteille on todella suositeltavaa, kun alat kirjoittaa pitempiä ohjelmia tai työskentelet yhdessä muiden ohjelmoijien kanssa. Yleisesti ottaen nimeämissääntöjä kannattaa noudattaa, jos lomakkeessa on enemmän kuin kaksi samantyyppistä objektia.

Jos haluat harjoitella ohjelmointitaitojasi

Lue asiakirja Review & Practice 3, joka on tallennettu kansion \Vb5Sbs\Review tiedostoon Rev3.doc. Se sisältää vaiheittain eteneviä tehtäviä, joilla voit testata ohjelmointitaitosi ja kerrata lukujen 7 ja 8 sisällön. Tehtävissä muodostetaan ohjelma, jolla etsitään, verrataan ja tulostetaan bittikarttatiedostoja.

Jos haluat jatkaa seuraavaan harjoitukseen

➤ Jätä Visual Basic käyntiin ja siirry harjoitukseen 9.

Jos haluat lopettaa Visual Basicin

➤ Valitse File-valikosta komento Exit.

Jos näyttöön tulee Save-valintaikkuna, napsauta Yes.

Harjoituksen tiivistelmä

Tehtävä	Ohje	Painike
Piirtää lomakkeeseen suoria viivoja	Käytä Viiva-työkalua.	
Piirtää lomakkeeseen suorakulmioita, soikioita ja ympyröitä	Käytä Kuvio-työkalua. Määritä objektin tyyppi ja piirteet Shape-ominaisuudella.	
Tehdä graafisia komentopainikkeita	Lisää lomakkeeseen kuvakehyksiä ja lataa niihin bittikarttakuvia. Kirjoita hiiren napsautuksia käsittelevät lauseet kuvakehysten MouseDown-aliohjelmiin.	
Lisätä ohjelmaan vedon ja pudotuksen tuki	Salli objektin veto ja pudotus asettamalla sen DragMode-ominaisuudeksi 1. Voit valita objektia varten vetokuvakkeen. Kirjoita DragDrop- tai DragOver-aliohjelma kohdeobjektiin, johon lähdeobjekti vedetään tai pudotetaan.	

Tehtävä	Ohje
Valita osoittimelle valmis muoto	Sijoita lomakkeen tai jonkin sen objektin MousePointer-ominaisuuteen valmiin osoittimen koodinumero.
Määrittää oma osoitin	Määritä MousePointer-ominaisuudeksi 99 ja määritä oma osoitin MouseIcon-ominaisuudella.
Siirtää objektia lomakkeessa	Siirrä objektia Move-metodilla. Esimerkki: `Picture1.Move 1440, 1440`
Animoida objekti	Sijoita yksi tai useita Move-metodeja ajastimen tapahtuma-aliohjelmaan. Animaation nopeus riippuu ajastimen Interval-ominaisuudesta.
Suurentaa tai pienentää objektia ohjelmallisesti	Muuta objektin Height- tai Width-ominaisuutta.
Nimetä objekti	Määritä nimi Name-ominaisuudella. Noudata vakiintuneita nimeämissääntöjä, jotta objektin tunnistaminen helpottuu.

Kun tarvitset tietoja aiheesta	Valitse Visual Basicin Help-valikosta Microsoft Visual Basic Help Topics, valitse Index-välilehti ja...
Viiva-työkalu	Kirjoita **Line**.
Kuvio-työkalu	Kirjoita **Shape**.
Grafiikkametodit	Kirjoita **graphics methods**.
Graafisten komento-painikkeiden luominen	Kirjoita **Image**.
Veto ja pudotus	Kirjoita **Drag**.
Objektin siirtäminen	Kirjoita **Move method**.

Tietoja seuraavasta harjoituksesta

Seuraavassa harjoituksessa opit luomaan ohjelmaan vakiomoduulin ja lisäämään siihen julkisia muuttujia, funktioita ja aliohjelmia. Siinä kuvattavista taidoista on hyötyä varsinkin suurehkoissa ohjelmointiprojekteissa ja ryhmätyössä.

Osa

4

Laajemmat ohjelmat

Harjoitus 9
Moduulien ja aliohjelmien käyttö 249

Harjoitus 10
Kokoelmien ja taulukoiden käsittely 275

Harjoitus 11
Tekstitiedostojen ja tietokantojen käsittely 301

Harjoitus 12
Yhteydet Microsoft Officeen 331

Harjoitus

9

Moduulien ja aliohjelmien käyttö

Arvioitu aika
55 min.

Tässä harjoituksessa opit

- luomaan vakiomoduuleja
- luomaan omia julkisia muuttujia ja aliohjelmia
- kutsumaan julkisia muuttujia ja aliohjelmia tapahtuma-aliohjelmista.

Kun olet tutkinut ja kirjoittanut harjoituksissa 1–8 esitellyt ohjelmat, tiedät jo aika paljon Visual Basicista. Olet oppinut Visual Basic -ohjelmoinnin perusteet, ja sinulla on taidot, jotka riittävät monenlaisten hyödyllisten apuohjelmien tekemiseen. Osassa 4 näet, mitä laajempien ohjelmien laatiminen vaatii. Kerron ensiksi, miten muodostetaan vakiomoduuleja.

Vakiomoduuli on ohjelman erillinen osa, joka sisältää yleisiä eli *julkisia* muuttujia ja aliohjelmia. Tässä harjoituksessa opit luomaan omia julkisia muuttujia ja aliohjelmia sekä kutsumaan niitä tapahtuma-aliohjelmista. Nämä taidot ovat erityisen tärkeitä suurissa ohjelmointiprojekteissa ja ryhmätyössä.

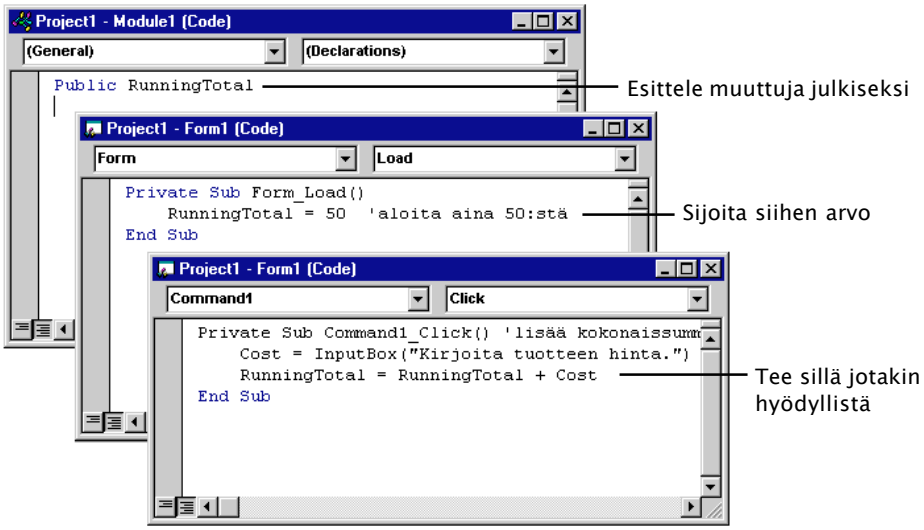
Vakiomoduulien käsittely

Kun kehittelet laajoja ohjelmia, sinulla on todennäköisesti useita lomakkeita ja tapahtuma-aliohjelmia, jotka käyttävät samoja muuttujia ja aliohjelmia. Muuttujat ovat normaalisti *paikallisia* tapahtuma-aliohjelmassa eli niitä voidaan lukea tai muuttaa ainoastaan siinä tapahtuma-aliohjelmassa, jossa ne on luotu. Vastaavasti tapahtuma-aliohjelmat ovat paikallisia lomakkeessa — et esimerkiksi voi kutsua Form1-lomakkeen cmdQuit_Click-aliohjelmaa Form2-lomakkeesta.

Vakiomoduulien avulla jaetaan muuttujia ja aliohjelmia ohjelman kaikkien osien kesken.

Jos muuttujien ja aliohjelmien pitää olla yhteiset projektin kaikille lomakkeille ja tapahtuma-aliohjelmille, sinun on esiteltävä ne projektin yhdessä tai useassa vakiomoduulissa. *Vakiomoduuli* eli koodimoduuli on erikoistiedosto, jonka tarkenne on *.bas* ja joka sisältää ohjelman kaikille osille yhteisiä muuttujia ja aliohjelmia. Lomakkeiden tapaan vakiomoduulit on lueteltu erikseen Project-ikkunassa. Ne tallennetaan levylle File-valikon komennolla Save Module1 As. Lomakkeista poiketen vakiomoduulit eivät sisällä objektien tai ominaisuuksien määrityksiä, vaan ainoastaan lauseita, joita voidaan katsella ja muokata Code-ikkunassa.

Seuraavasta kuvasta näkyy, miten vakiomoduulissa esiteltyä julkista muuttujaa voidaan käyttää Visual Basic -projektin muissa tapahtuma-aliohjelmissa.



HUOMAA Lomakkeeseen liittyvät objektit ja tapahtuma-aliohjelmat tallennetaan *lomakemoduuliin*, ja uusi objekti muodostetaan *luokkamoduulissa*.

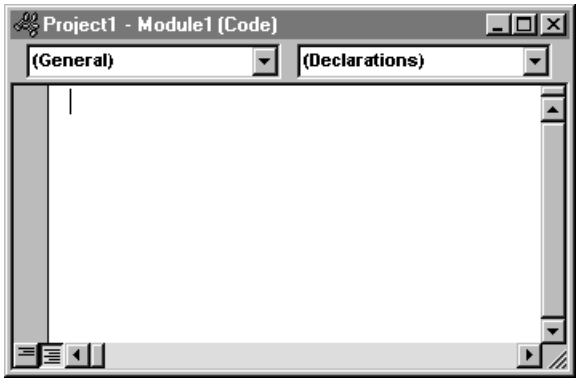
Vakiomoduulin luominen

Kun aiot tehdä uuden vakiomoduulin, napsauta työkalurivin Lisää moduuli -painikkeen alanuolta ja valitse Module tai valitse Project-valikosta komento Add Module. Luomasi vakiomoduuli näkyy heti Code-ikkunassa. Ohjelman ensimmäisen vakiomoduulin oletusnimi on Module1, mutta voit vaihtaa sen tallentaessasi moduulin. Kokeile nyt tyhjän vakiomoduulin lisäämistä projektiin.

Vakiomoduulin luominen ja tallentaminen

1 Käynnistä Visual Basic, valitse Project-valikosta komento Add Module ja napsauta Open.

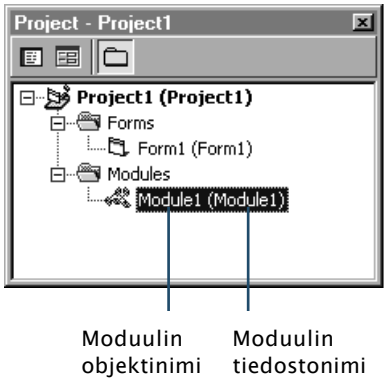
Visual Basic lisää projektiin vakiomoduulin nimeltä Module 1. Se näkyy Code-ikkunassa tällaisena:



Ikkunan yläreunan objekti- ja aliohjelmaluetteloista ilmenee, että vakiomoduulin yleisten esittelyiden osa on avoinna. Tässä esitellyt muuttujat ja aliohjelmat ovat koko ohjelman käytettävissä. (Voit kohta kokeilla muuttujien ja aliohjelmien esittelemistä.)

2 Kaksoisnapsauta Project-ikkunan otsikkoriviä, jotta näet koko ikkunan.

Project-ikkuna muuttuu tällaiseksi:



Project-ikkuna näyttää ohjelmaan lisäämäsi vakiomoduulin uudessa kansiossa. Sulkein ympäröity nimi Module1 ilmaisee moduulitiedoston oletusnimen. Sulkeiden vasemmalla puolella näkyy sana Module eli moduulin nimi ohjelmassa. Vaihdat molemmat asetukset seuraavissa vaiheissa.

3 Valitse File-valikosta komento Save Module1 As, jolla tallennetaan tyhjä vakiomoduuli levylle.

Vakiomoduuli-tiedostojen tarkenne on .bas.

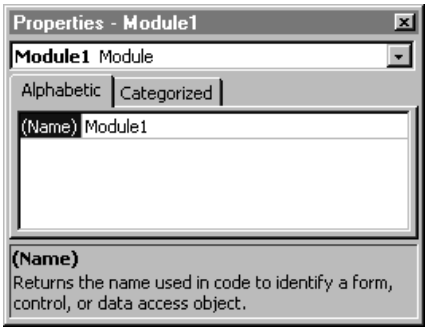
4 Valitse kansio \Vb5Sbs\Less09. Kirjoita **MyTestMod.bas** ja paina ENTER.

Vakiomoduuli tallentuu levylle .bas-tiedostoon, ja Project-ikkunassa näkyvä moduulitiedoston nimi vaihtuu.

HUOMAA Voit ladata tämän tiedoston toiseen projektiin Project-valikon komennolla Add File.

5 Kaksoisnapsauta Properties-ikkunan otsikkoriviä.

Properties-ikkuna suurenee täysikokoiseksi, kuten seuraavasta kuvasta näkyy:



Koska vakiomoduulissa ei ole objekteja, sen ainoa ominaisuus on Name. Name-ominaisuudella annetaan moduulille objektinimi, jonka perusteella moduulit erottuvat toisistaan. Nimeämissääntöjen mukaan moduulien nimissä on etuliite *mod*.

Name-ominaisuudella määritetään moduulin objektinimi.

6 Vaihda Name-ominaisuudeksi **modVariables** ja paina ENTER.

Vakiomoduulin objektinimi muuttuu Properties-, Project- ja Code-ikkunoissa.

Kuten huomaat, projektin vakiomoduuleja käsitellään suunnilleen samalla tavoin kuin lomakkeita. Seuraava tehtäväsi on lisätä luomaasi vakiomoduuliin julkinen muuttuja.

HUOMAA Jos haluat poistaa vakiomoduulin projektista, valitse moduuli Project-ikkunasta ja valitse Project-valikosta komento Remove. Tämä komento ei poista moduulia kiintolevyltä, vaan katkaisee linkin valitun moduulin ja nykyisen projektin väliltä.

Julkisten muuttujien käsittely

Yleisen eli julkisen muuttujan esitteleminen vakiomoduulissa on yksinkertaista: kirjoita avainsana *Public* ja muuttujan nimi. Kun olet esitellyt muuttujan, sen arvo voidaan lukea, muuttaa tai näyttää ohjelman missä tahansa aliohjelmassa. Esimerkiksi lause

```
Public Loppusumma
```

esittelee vakiomoduulissa julkisen muuttujan nimeltä Loppusumma.

Julkiset muuttujat ovat ohjelman kaikkien aliohjelmien käytettävissä.

Julkiset muuttujat ovat normaalisti muunnelmatyyppisiä, mutta voit valita perustietotyypin As-avainsanalla. Esimerkiksi lause

```
Public Sukunimi As String
```

esittelee julkisen merkkijonomuuttujan nimeltä Sukunimi.

Lucky Seven on harjoituksesta 1 tuttu peliohjelma.

Seuraava ohjelma havainnollistaa julkisen Wins-nimisen muuttujan käyttämistä vakiomoduulissa. Tutkimme jälleen Onnen seiska -ohjelmaa eli ensimmäistä tässä kirjassa kirjoittamaasi ohjelmaa. Win-muuttujaan tallennetaan tieto siitä, kuinka monta peliä olet voittanut.

Palaaminen Onnen seiska -projektiin

1. Napsauta työkalurivin Avaa projekti -painiketta, hylkää tekemäsi muutokset napsauttamalla No ja avaa kansiosta \Vb5Sbs\Less01 projekti Lucky.vbp.
2. Ellei Lucky-lomaketta näy, valitse Project-ikkunasta Lucky.frm ja napsauta Näytä objekti -painiketta. (Muuta lomakeikkunan kokoa tarvittaessa.)

   Näet tällaisen käyttöliittymän:

Käynnistä-painike

3. Käynnistä ohjelma napsauttamalla työkalurivin Käynnistä-painiketta.
4. Napsauta Pyöritä-painiketta 6 tai 7 kertaa ja sitten End-painiketta.

   Voitat viisi ensimmäistä peliä (yksi numeroista on aina 7), mutta sitten onnesi kääntyy. Kuten muistanet, ohjelma tuottaa Rnd-funktiolla kolme satunnaislukua aina, kun napsautat Pyöritä-painiketta. Jos yksi numeroista on 7, Spin-painikkeen tapahtuma-aliohjelma (Command1_Click) näyttää kolikkopinon ja antaa äänimerkin.

   Tehtäväsi on lisätä lomakkeeseen uusi teksti ja julkinen muuttuja, joka pitää kirjaa voittojesi määrästä.
5. Valitse File-valikosta komento Save Lucky.frm As. Valitse kansio \Vb5Sbs\Less09 ja tallenna lomake nimellä **MyWins.frm**.
6. Valitse File-valikosta komento Save Project As. Valitse kansio \Vb5Sbs\Less09 ja tallenna projekti nimellä **MyWins.vbp**.

Tee uusi ohjelma muokkaamalla Wins-lomaketta ja lisäämällä vakiomoduuli.

Vakiomoduulin lisääminen

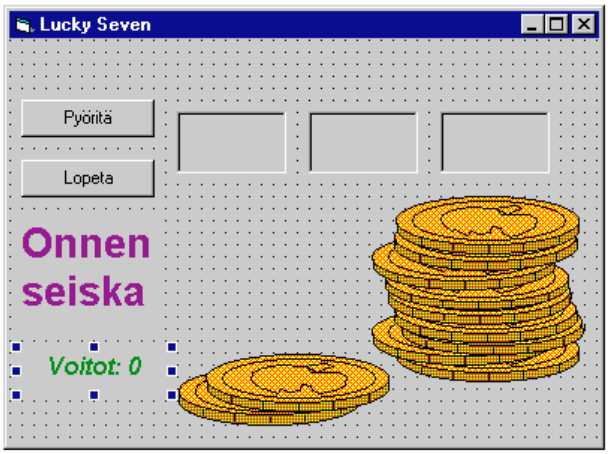
1. Pienennä ohjelman nimen sisältävää tekstiobjektia, jotta se vie vähemmän tilaa lomakkeessa. Objekti ulottuu nyt pitkälle Onnen seiska -tekstin alapuolelle.
2. Napsauta Teksti-työkalua ja piirrä uusi suorakulmainen tekstiobjekti entisen tekstiobjektin alapuolelle.

Teksti-työkalu

3. Määritä uuden tekstiobjektin ja lomakkeen ominaisuudet seuraavan taulukon mukaisesti. Vaihda uuden tekstiobjektin nimeksi lblWins, jotta se on helppo tunnistaa.

Objekti	Ominaisuus	Asetus
Label5	Alignment	2 - Center
	Caption	”Voitot: 0”
	Font	Arial, Bold Italic, 12 pistettä
	ForeColor	Green
	Name	lblWins
Form1	Caption	”Onnen seiska”

Valmiin lomakkeen pitäisi näyttää seuraavan sivun kuvan mukaiselta.



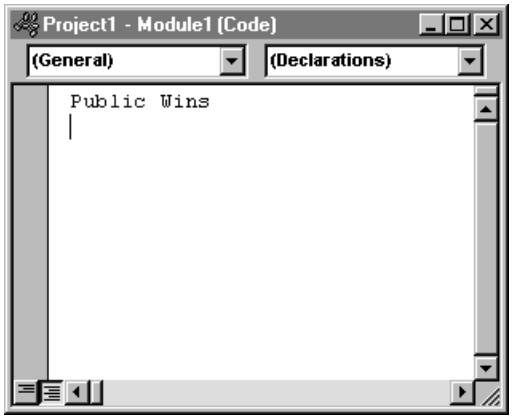
Lisää seuraavaksi projektiin uusi vakiomoduuli.

4 Valitse Project-valikosta komento Add Module ja napsauta Open.

Moduuli nimeltä Module1 ilmestyy Code-ikkunaan.

5 Kirjoita vakiomoduuliin **Public Wins** ja paina ENTER.

Tämä ohjelmalause esittelee muunnelmatyyppisen julkisen muuttujan. Kun ohjelma on käynnissä, sen jokainen tapahtuma-aliohjelma voi viitata tähän muuttujaan. Vakiomoduulin pitäisi näyttää tällaiselta:



6 Valitse File-valikosta komento Save Module1 As, kirjoita **MyWins.bas** ja paina ENTER, jotta moduuli tallentuu levylle.

7 Valitse Project-ikkunasta Form1 (MyWins.frm), napsauta Näytä objekti -painiketta ja kaksoisnapsauta Pyöritä-painiketta.

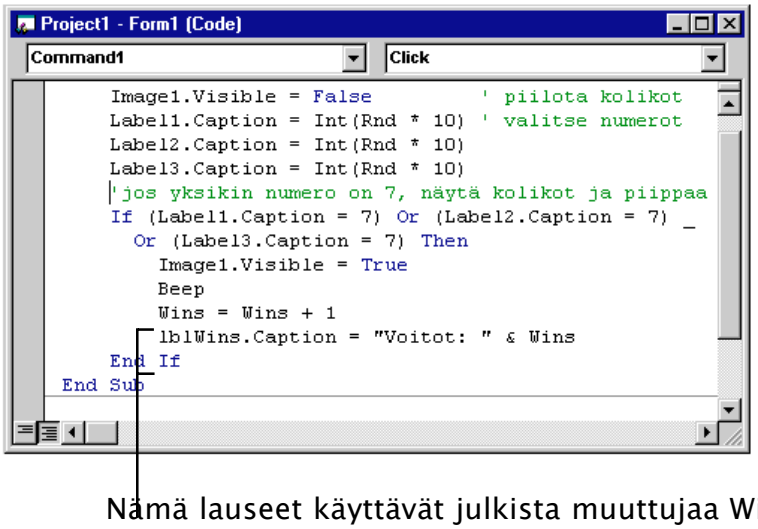
Pyöritä-painikkeen tapahtuma-aliohjelma Command1_Click ilmestyy Code-ikkunaan.

8 Kirjoita tapahtuma-aliohjelmaan seuraavat lauseet Beep-lauseen alapuolelle:

```
Wins = Wins + 1
lblWins.Caption = "Voitot:  " & Wins
```

Ensimmäinen lause suurentaa julkisen Wins-muuttujan arvoa yhdellä, jos yksi numeroista on 7. Toinen lause käyttää yhteenliittämisoperaattoria (&) ja sijoittaa lblWins-objektiin merkkijonon *Voitot: X*, jossa *X* on voittojen määrä. Valmiin tapahtuma-aliohjelman pitäisi olla tällainen:

Julkinen muuttuja Wins päivitetään tapahtuma-aliohjelmassa.



9 Sulje Code-ikkuna ja tallenna projekti napsauttamalla Tallenna projekti -painiketta.

10 Käynnistä ohjelma napsauttamalla Käynnistä-painiketta.

11 Napsauta Pyöritä-painiketta kymmenen kertaa.

Voittojesi määrä näkyy Wins-tekstin jäljessä. Joka kerta voittaessasi määrä kasvaa yhdellä. Kymmenen kierroksen jälkeen olet voittanut kuudesti, kuten seuraavan sivun kuvasta näkyy.

12 Lopeta ohjelma napsauttamalla Lopeta-painiketta.

Julkinen Wins-muuttuja oli hyödyllinen tässä ohjelmassa, koska se säilytti arvonsa Command1_Click-aliohjelman kutsujen välilläkin. Jos olisit esitellyt Wins-muuttujan paikallisesti Command1_Click-aliohjelmassa, se olisi nollautunut joka kierroksella. Nollautuminen estetään esittelemällä julkinen muuttuja vakiomoduulissa.

Wins-muuttuja pitää kirjaa voitoistasi.

Yleiskäyttöisten aliohjelmien esitteleminen

Vakiomoduuleissa voi olla julkisten muuttujien lisäksi yleiskäyttöisiä aliohjelmia, joita voidaan kutsua ohjelman mistä tahansa osasta. Yleiskäyttöiset aliohjelmat eroavat tapahtuma-aliohjelmista, koska ne eivät liity ajonaikaisiin tapahtumiin eivätkä työkaluilla luotuihin objekteihin. Ne muistuttavat Visual Basicin valmiita lauseita ja funktioita: niitä kutsutaan nimellä, ne voivat saada argumentteja ja ne suorittavat tietyn tehtävän.

Kuvittele esimerkiksi ohjelma, jossa on kolme tapaa tulostaa bittikartta: valikkokomento Tulosta, työkalurivin Tulosta-painike sekä tulostinkuvake, johon bittikartta voidaan vetää ja pudottaa. Voisit kirjoittaa samat tulostuslauseet jokaiseen kolmeen tapahtuma-aliohjelmaan tai hoitaa kaikista lähteistä tulevat tulostuspyynnöt vakiomoduulin yhdellä aliohjelmalla. Yleiskäyttöiset aliohjelmat vähentävät kirjoitustyötä, tekevät ohjelmasta pienemmän ja helpommin hallittavan sekä muuttavat tapahtuma-aliohjelmat helpommin ymmärrettäviksi.

Voit luoda vakiomoduuliin kolmenlaisia yleiskäyttöisiä aliohjelmia:

Vakiomoduulin funktio- ja lausealiohjelmat toimivat yleiskäyttöisinä aliohjelmina.

Funktioaliohjelmat Funktioaliohjelmia kutsutaan nimellä tapahtuma- ja muista aliohjelmista. Ne voivat saada argumentteja, ja ne palauttavat aina arvon funktion nimessä. Niitä käytetään yleensä laskutoimituksissa.

Lausealiohjelmat Lausealiohjelmia kutsutaan nimellä tapahtuma- ja muista aliohjelmista. Ne voivat saada argumentteja, joita käytetään lauseissa ja joiden kautta palautetaan arvoja. Funktiosta poiketen lausealiohjelma ei palauta arvoa nimessään. Lausealiohjelmia käytetään yleensä tietoja luettaessa, muokattaessa ja näytettäessä sekä ominaisuuksia määritettäessä.

Ominaisuusaliohjelmat Ominaisuusaliohjelmilla luodaan ja käsitellään käyttäjän määrittämiä ominaisuuksia. Ne ovat kätevä, joskin melko vaativa tapa mukauttaa Visual Basicin työkaluja ja laajentaa Visual Basic -kieltä. Jos haluat lisätietoja, etsi ohjeista hakusanaa *property get statement*.

Yleiskäyttöisten aliohjelmien etuja

Voit kirjoittaa usein tarvittavat ohjelmalauseet vakiomoduuliin ja antaa niille helposti tunnistettavan nimen. Näin muodostettavilla yleiskäyttöisillä aliohjelmilla on seuraavia etuja:

- Samoja lauseita ei tarvitse toistaa. Kun olet kerran määrittänyt aliohjelman, sitä voidaan kutsua miten monta kertaa tahansa.
- Ohjelmasta tulee helppolukuisempi. Ohjelma, joka on jaettu useaan pieneen osaan, on helpompi hahmottaa ja ymmärtää kuin suuri yksiosainen ohjelma.
- Ohjelman kehittäminen yksinkertaistuu. Loogisista yksiköistä koostuvat ohjelmat ovat helpompia suunnitella, kirjoittaa ja testata. Jos toimit työryhmän jäsenenä, voit vaihtaa heidän kanssaan aliohjelmia ja moduuleja kokonaisten ohjelmien sijasta.
- Uudelleenkäyttö helpottuu. Voit helposti sisällyttää vakiomoduulien aliohjelmia muihin ohjelmointiprojekteihin.
- Visual Basic -kieli laajenee. Aliohjelmat pystyvät usein suorittamaan tehtäviä, joihin Visual Basicin yksittäiset lauseet eivät riitä.

Funktioaliohjelmien kirjoittaminen

Funktio suorittaa tehtävän, esimerkiksi laskutoimituksen, ja palauttaa arvon.

Funktioaliohjelma eli funktio on joukko lauseita, jotka on sijoitettu vakiomoduuliin Function-lauseen ja End Function -lauseen väliin. Funktion lauseet hoitavat tarvittavat tehtävät, esimerkiksi käsittelevät tekstiä, muokkaavat syöttötietoja tai laskevat lukuarvon. Funktio suoritetaan eli sitä *kutsutaan* kirjoittamalla sen nimi ja mahdolliset pakolliset argumentit ohjelmalauseeseen. (*Argumentit* sisältävät tietoja,

joita funktio käsittelee.) Toisin sanoen funktioaliohjelmaa käytetään täsmälleen samalla tavoin kuin Visual Basicin valmiita funktioita, kuten Time, Int tai Str.

HUOMAA Vakiomoduuleissa määritetyt funktiot ovat normaalisti julkisia: niitä voidaan kutsua mistä tahansa tapahtuma-aliohjelmasta.

Funktion muoto

Funktion perusmuoto on tällainen:

```
Function FunktionNimi([argumentit]) [As Tyyppi]
    funktion lauseet
End Function
```

Funktiolla voi olla tyyppi. Valinnaiset osat on ympäröity hakasulkein ([]). Muut osat ovat pakollisia.

Seuraavat osat ovat tärkeitä:

- *FunktionNimi* on funktioaliohjelman nimi.
- *Argumentit* on funktiossa käytettävien (pilkuin eroteltujen) argumenttien luettelo. Se ei ole pakollinen.
- *Tyyppi* on valinnainen määritys, joka ilmaisee funktion palautustyypin (oletustyyppi on muunnelma eli Variant).
- *Funktion lauseet* on lauselohko, joka suorittaa funktiolle annetun tehtävän.

Funktio palauttaa kutsuvaan aliohjelmaan aina arvon, joka on tallennettu funktion nimeen (*FunktionNimi*). Siksi funktion viimeinen lause on usein sijoituslause, joka sijoittaa laskutoimituksen tuloksen funktion nimeen. Esimerkiksi seuraava Kokonaisvero-funktio laskee tuotteelle arvonlisäveron ja valmisteveron ja sijoittaa tuloksen muuttujaan Kokonaisvero:

```
Function Kokonaisvero(Hinta)
    ALV = Hinta * 0.22          'arvonlisävero on 22 %
    Valmistevero = Hinta * 0.06 ' valmistevero on 6 %
    Kokonaisvero = ALV + Valmistevero
End Function
```

Kokonaisvero on esimerkki yksiargumenttisesta funktiosta.

TÄRKEÄÄ Suosittelen, että sijoitat funktion nimeen arvon aina funktiota kirjoittaessasi. Näin voit olla varma siitä, mikä tulos ohjelmaan palautuu.

Funktion kutsuminen

Voisit kutsua Kokonaisvero-funktiota tapahtuma-aliohjelmasta tällaisella lauseella:

```
lblVerot.Caption = Kokonaisvero(500)
```

Funktion arvo sijoitetaan yleensä muuttujaan tai ominaisuuteen.

Tämä lause laskee verot 500 markan hintaiselle tuotteelle ja sijoittaa tuloksen lblVerot-objektin Caption-ominaisuuteen. Kokonaisvero-funktio voi saada argumentikseen myös muuttujan, kuten tästä lauseesta näkyy:

```
Kokonaishinta = Myyntihinta + Kokonaisvero(Myyntihinta)
```

Tämä lause laskee Kokonaisvero-funktiolla myyntihintaa vastaavat verot ja laskee ne yhteen myyntihinnan kanssa kokonaishinnaksi. Huomaat varmaan, kuinka funktio tekee ohjelmasta paljon selkeämmän.

Laskutoimituksen suorittaminen funktiolla

Kirjoita seuraavaksi Onnen seiska -ohjelmaan funktio, joka laskee voittojen suhteellisen määrän prosentteina. Tätä varten sinun pitää lisätä vakiomoduuliin funktio nimeltä Rate ja julkinen muuttuja nimeltä Spins. Ohjelman tulee kutsua Rate-funktiota aina, kun Pyöritä-painiketta napsautetaan. Tulos näytetään uudessa lomakkeeseen lisättävässä tekstiobjektissa.

Voittoprosenttifunktion luominen

1. Avaa Project-ikkuna.

   MyWins.vbp-projektin osat näkyvät Project-ikkunassa. Tallenna ne nimellä MyRate, jotta alkuperäinen MyWins-ohjelma säilyy ennallaan.

2. Napsauta MyWins.frm-lomaketta. Valitse File-vailkosta komento Save MyWins.frm As. Tallenna lomake kansioon \Vb5Sbs\Less09 nimellä **MyRate.frm**.

3. Napsauta Project-ikkunan vakiomoduulia MyWins.bas. Valitse File-valikosta komento Save MyWins.bas As. Tallenna moduuli nimellä **MyRate.bas**.

4. Valitse File-valikosta komento Save Project As. Tallenna projekti nimellä **MyRate.vbp**.

5. Ellet näe lomaketta, valitse Project-ikkunasta MyRate.frm-lomake ja napsauta Näytä objekti -painiketta.

   Näytä objekti -painike

   Onnen seiska -ohjelman käyttöliittymä tulee näkyviin.

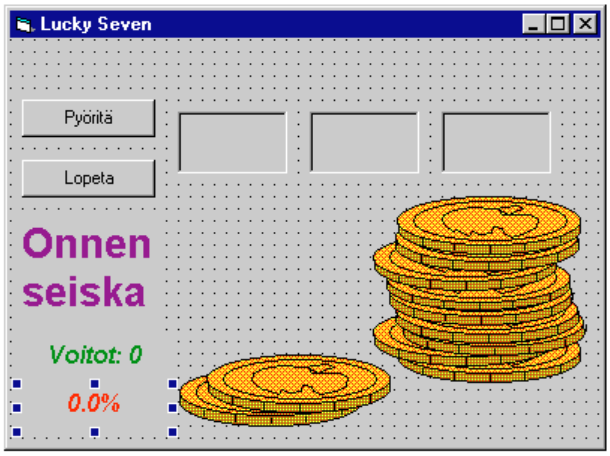
6. Siirrä Wins-tekstiobjekti ylemmäksi, jotta saat tilaa uudelle tekstiobjektille. Sinun pitää ehkä pienentää kumpaakin nykyistä objektia.

Teksti-työkalu

7 Napsauta Teksti-työkalua ja lisää uusi tekstiobjekti Wins-tekstiobjektin alapuolelle. Määritä sen ominaisuudet näin:

Objekti	Ominaisuus	Asetus
Label5	Alignment	2 - Center
	Caption	”0.0%”
	Font	Arial, Bold Italic, 12 pistettä
	ForeColor	Red
	Name	lblRate

Lomakkeen pitäisi näyttää tällaiselta:



8 Valitse Project-ikkunasta MyRate.bas-moduuli ja napsauta Näytä lauseet -painiketta.

Module1-vakiomoduuli ilmestyy Code-ikkunaan.

9 Kirjoita Public Wins -lauseen alapuolelle seuraava julkisen muuttujan esittely:

```
Public Spins
```

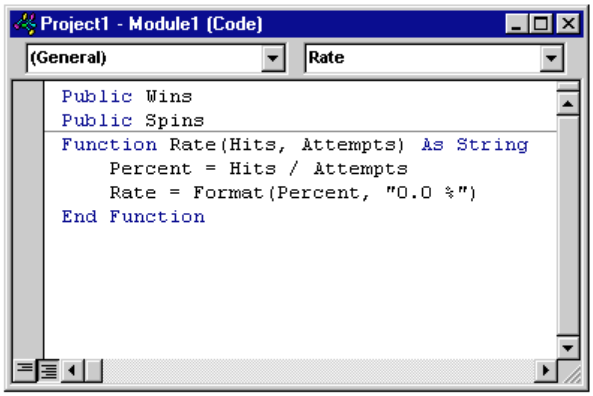
Vakiomoduulissa on nyt kaksi julkista muuttujaa (Wins ja Spins), jotka ovat ohjelman kaikkien aliohjelmien käytettävissä. Spins-muuttujaan on tarkoitus tallentaa pelattujen pelien määrä.

10 Kirjoita tällainen funktio:

```
Function Rate(Hits, Attempts) As String
    Percent = Hits / Attempts
    Rate = Format(Percent, "0.0%")
End Function
```

Rate-funktio sijoitetaan Module1-vakiomoduuliin.

Kun olet kirjoittanut funktion ensimmäisen rivin, Visual Basic avaa vakiomoduuliin uuden aliohjelman, johon voit lisätä funktion sisällön. Funktion loppuosan kirjoitettuasi Project-ikkunan pitäisi olla samanlainen kuin seuraava kuva:



Rate-funktio laskee voittosuhteen jakamalla Hits-argumentin Attempts-argumentilla ja muokkaa tulosta Format-funktiolla. Rate-funktio on esitelty merkkijonotyyppiseksi, koska Format-funktio palauttaa merkkijonon. Hits- ja Attempts-argumentti ovat paikkamerkkejä kahdelle muuttujalle, jotka välitetään funktiolle kutsussa. Funktiolle voidaan välittää mitä tahansa lukuja tai muuttujia, ei ainoastaan Wins ja Spins.

11 Sulje Code-ikkuna ja kaksoisnapsauta lomakkeen Pyöritä-painiketta, jotta tapahtuma-aliohjelma Command1_Click tulee esiin.

12 Lisää aliohjelman neljännen lauseen (jossa on Rnd-funktio) alapuolelle tällainen lause:

```
Spins = Spins + 1
```

Lause suurentaa Spins-muuttujan arvoa yhdellä, kun käyttäjä napsauttaa Pyöritä-painiketta, ja pelin ruutuihin ilmestyy uudet numerot.

13 Vieritä Code-ikkunaa alaspäin ja kirjoita Command1_Click-aliohjelman loppuun, End If- ja End Sub- lauseiden väliin, seuraava lause:

```
lblRate.Caption = Rate(Wins, Spins)
```

Tässä funktion kutsussa on kaksi muuttujaa.

Kun kirjoitat Rate-funktion kutsun, huomaa, miten Visual Basic näyttää automaattisesti funktion argumenttien nimet muistin tueksi.

Tämä lause kutsuu Rate-funktiota ja antaa sille argumentteina muuttujat Wins ja Spins. Funktio palauttaa tuloksen merkkijonomuotoisena prosenttilukuna. Ohjelma sijoittaa tuloksen lblRate-tekstiobjektin Caption-ominaisuuteen jokaisen pelin jälkeen. Siinä olikin kaikki!

Tallenna projekti -painike

14 Sulje Code-ikkuna ja päivitä projektitiedostot napsauttamalla Tallenna projekti -painiketta.

Kokeile ohjelmaa.

MyRate-ohjelman suorittaminen

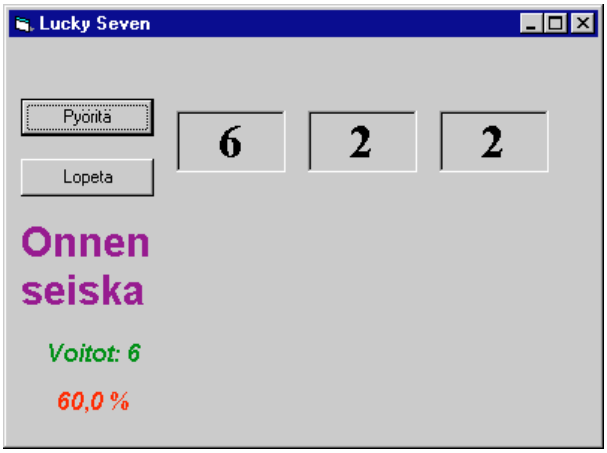
Käynnistä-painike

1 Käynnistä ohjelma napsauttamalla Käynnistä-painiketta.

2 Napsauta Pyöritä-painiketta kymmenen kertaa.

Viidellä ensimmäisellä kerralla voittosuhde on 100.0 % eli voitat joka pelissä. Kun jatkat Pyöritä-painikkeen napsauttelemista, voittosuhde vaihtelee: 83.3 %, 71.4 %, 75.0 % (taas voitto), 66.7 % ja 60.0 % (yhteensä 6 voittoa 10 pelistä). Lopulta lomake näyttää tällaiselta:

Valmis Rate-ohjelma on kansiossa \Vb5Sbs\Less09.



Onnen seiska -ohjelman todellinen voittosuhde on noin 28 %.

Jos edelleen jatkat pelaamista, voit huomata, että voittosuhde pienenee noin 28 prosenttiin. Rate-funktiosta näet, että olit todella onnekas ensimmäisillä kierroksilla, mutta sitten karu todellisuus tuli esiin.

3 Kun olet tarpeeksi pelannut, napsauta End-painiketta.

Ohjelman suoritus päättyy, ja ohjelmointiympäristö palaa näkyviin.

HUOMAA Jos haluat muokata ohjelmaa niin, että se tuottaa aina käynnistämisen jälkeen eri satunnaislukusarjan, lisää Form_Load-aliohjelmaan Randomize-lause. Ohjeet ovat harjoituksen 1 kohdassa "Lisätietoja: Lisäysten tekeminen ohjelmaan".

Lausealiohjelmien kirjoittaminen

Lausealiohjelmat käsittelevät tietoja.

Lausealiohjelma on samanlainen kuin funktioaliohjelma, mutta sen alussa on Sub-avainsana, ja se ei palauta arvoa nimessään. Lausealiohjelmalla yleensä pyydetään tietoja käyttäjältä, näytetään tai tulostetaan tietoja tai muokataan ominaisuuksia ehdon perusteella. Sillä voidaan myös käsitellä ja palauttaa useita muuttujia. Useimmat funktiot voivat palauttaa vain yhden arvon, mutta lausealiohjelma voi palauttaa useita arvoja käyttämällä muuttujia.

Lausealiohjelman muoto

Lausealiohjelman perusmuoto on tällainen:

```
Sub AliohjelmanNimi([argumentit])
    aliohjelman lauseet
End Sub
```

Seuraavat osat ovat tärkeitä:

- *AliohjelmanNimi* on lausealiohjelman nimi.
- *Argumentit* on lausealiohjelmassa käytettävien (pilkuin eroteltujen) argumenttien luettelo. Se ei ole pakollinen.
- *Aliohjelman lauseet* on lauselohko, joka suorittaa lausealiohjelmalle annetun tehtävän.

Aliohjelman kutsun argumenttien on vastattava esittelyssä mainittuja argumentteja.

Aliohjelman kutsun argumenttien määrän ja tyyppien on oltava täsmälleen samat kuin aliohjelman määrittelyssä. Jos aliohjelma muokkaa sille välitettyjä muuttujia, muutetut arvot palautuvat ohjelmaan. Vakiomoduulissa määritetyt aliohjelmat ovat oletuksen mukaan julkisia, joten niitä voidaan kutsua mistä tahansa tapahtuma-aliohjelmasta.

TÄRKEÄÄ Aliohjelmaan välitettyä muuttujaa sanotaan *viittaus-argumentiksi*, koska aliohjelma voi muokata muuttujaa ja palauttaa tämän kutsuvaan aliohjelmaan. Aliohjelmaan välitettyä vakioarvoa (esimerkiksi lainausmerkein ympäröityä merkkijonoa) taas sanotaan *arvoargumentiksi*, koska aliohjelma ei voi muokata sitä. Voit välittää muuttujiakin arvoina käyttämällä erikoismerkintää. Lisätietoja tästä on tämän luvun myöhemmässä kohdassa "Lisätietoja: Arvoargumentit".

Seuraava lausealiohjelma lisää nimiä lomakkeen luetteloruutuun. Se vastaanottaa yhden merkkijonomuuttujan viittausargumenttina. Jos tämä lausealiohjelma esitellään vakiomoduulissa, mikä tahansa ohjelman tapahtuma- aliohjelma voi kutsua sitä.

```
Sub AddNameToListBox(person$)
    If person$ <> "" Then
        Form1.List1.AddItem person$
        Msg$ = person$ & " lisätty   luetteloruutuun."
    Else
        Msg$ = "Nimi  puuttuu."
    End If
    MsgBox (Msg$), , "Lisää  nimi"
End Sub
```

Tämä lausealiohjelma saa argumentin person$.

AddNameToListBox-aliohjelma lukee luetteloon lisättävän nimen argumentista person$ eli viittausargumenttina välitetystä merkkijonomuuttujasta. Ellei merkkijonomuuttuja ole tyhjä, aliohjelma lisää annetun nimen LIst1-luettelo-ruutuobjektiin AddItem-metodilla ja ilmiottaa tästä MsgBox-funktiolla. Jos merkkijonomuuttuja on tyhjä, aliohjelma ohittaa AddItem-metodin ja ilmoittaa, että nimi puuttuu.

HUOMAA Kun määrität ominaisuuksia vakiomoduulin aliohjelmalla, sinun pitää lisätä objektin nimen edelle lomakkeen nimi (tässä esimerkissä Form1) ja piste. Näin Visual Basic tietää, mitä lomaketta tarkoitat.

Lausealiohjelman kutsuminen

Kun kutsut lausealiohjelmaa ohjelmassa, sinun pitää kirjoittaa sen nimi ja mahdolliset argumentit. Voisit kutsua edellä kuvattua AddNameToListBox-aliohjelmaa ja välittää sille vakiomerkkijonon (*arvoargumentin*) tällaisella lauseella:

Arvoparametreinä käytetään vakioarvoja.

```
AddNameToListBox "Kirsi"
```

Toisaalta voisit kutsua aliohjelmaa ja välittää sille muuttujan (*viittausargumentin*) näin:

Viittausparametreinä käytetään muuttujia.

```
AddNameToListBox NewName$
```

Kummassakin tapauksessa AddNameToListBox-aliohjelma lisää määritetyn nimen luetteloruutuun. Tässä tapauksessa arvo- ja viittausargumenteilla ei ole eroa, koska aliohjelma ei muuta argumentin arvoa.

Lausealiohjelman tilaa säästävät edut tulevat selvästi esiin, kun aliohjelmaa kutsutaan monta kertaa. Tästä on esimerkki seuraavalla sivulla.

```
AddNameToListBox "Kirsi"  ' lisää aina kaksi nimeä
AddNameToListBox "Raisa"
Do                        ' ja anna käyttäjän lisätä muita
    NewName$ = InputBox("Kirjoita luetteloon nimi.", "Lisää nimi")
    AddNameToListBox NewName$
Loop Until NewName$ = ""
```

Tässä käyttäjä voi lisätä luetteloruutuun niin monta nimeä kuin haluaa. Seuraavassa harjoituksessa voit harjoitella toisenlaisten syöttötietojen käsittelemistä lausealiohjelmalla.

Tietojen lukeminen lausealiohjelmalla

Lausealiohjelmilla käsitellään usein syöttötietoja, kun ne tulevat kahdesta tai useammasta lähteestä ja niiden pitää olla samassa muodossa. Seuraava tehtäväsi on kirjoittaa lausealiohjelma nimeltä AddName, joka kysyy käyttäjältä tietoja ja muotoilee tekstin niin, että se voidaan esittää tekstiruudun usealla rivillä. Tämä lausealiohjelma säästää ohjelmointiaikaa, koska käytät sitä kahdessa tapahtumaaliohjelmassa, jotka liittyvät eri tekstiruutuihin. Koska lausealiohjelma määritellään vakiomoduulissa, sinun pitää kirjoittaa se vain kerran.

Tekstiruudun lausealiohjelman luominen

1. Valitse File-valikosta komento New Project ja avaa vakiosovellus napsauttamalla OK.

   Uusi, tyhjä lomake tulee näkyviin.

2. Napsauta Tekstiruutu-työkalua ja piirrä lomakkeen keskelle kaksi vierekkäistä tekstiruutua.

   Tekstiruutu-työkalu

   Näihin tekstiruutuihin lisätään yrityksen kahden osaston työntekijöiden nimet.

3. Napsauta Teksti-työkalua ja lisää tekstiruutujen yläpuolelle kaksi tekstiobjektia.

   Teksti-työkalu

   Näihin objekteihin kirjoitetaan osastojen nimet.

4. Napsauta Komentopainike-työkalua ja lisää komentopainike kummankin tekstiruudun alapuolelle ja vielä kolmas painike lomakkeen alaosaan.

   Komentopainike-työkalu

   Kahdella ensimmäisellä komentopainikkeella lisätään työntekijöiden nimiä osastokohtaisiin tekstiruutuihin ja kolmannella komentopainikkeella lopetetaan ohjelma.

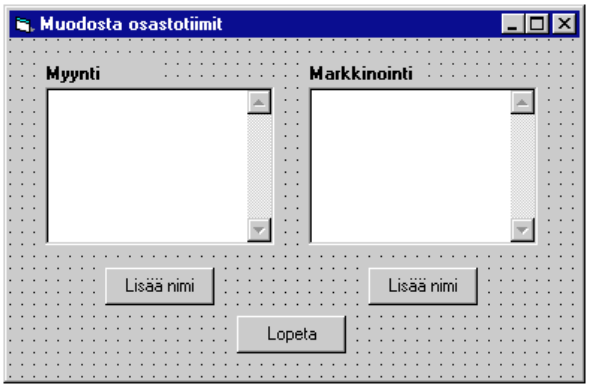
5 Määritä objektien ominaisuudet seuraavan taulukon mukaisesti.

Koska tekstiruutuihin kirjoitetaan useita rivejä, aseta niiden MultiLine-ominaisuudeksi True. Lisää pystyvierityspalkki asettamalla ScrollBars-ominaisuudeksi Vertical. Estä tekstiruutujen valitseminen TabStop-ominaisuudella False ja lukitse tekstiruudut Locked-ominaisuudella True, jotta ruutujen sisältöä ei voi muuttaa. Tällaisia asetuksia käytetään useimmiten monirivisissä tekstiruuduissa.

Nämä ovat tyypillisiä asetuksia tekstiruudulle, jossa näytetään monta tekstiriviä.

Objekti	Ominaisuus	Asetus
Text1	Text	(Tyhjä)
	MultiLine	True
	ScrollBars	2 - Vertical
	TabStop	False
	Locked	True
	Name	txtSales
Text2	Text	(Tyhjä)
	MultiLine	True
	ScrollBars	2 - Vertical
	TabStop	False
	Locked	True
	Name	txtMkt
Label1	Caption	”Myynti”
	Font	Bold
	Name	lblSales
Label2	Caption	”Markkinointi”
	Font	Bold
	Name	lblMkt
Command1	Caption	”Lisää nimi”
	Name	cmdSales
Command2	Caption	”Lisää nimi”
	Name	cmdMkt
Command3	Caption	”Lopeta”
	Name	cmdQuit
Form1	Caption	”Muodosta osastotiimit

Valmiin lomakkeen pitäisi näyttää tällaiselta:



Lisää seuraavaksi vakiomoduuli ja yleiskäyttöinen lausealiohjelma AddName.

6 Valitse Project-valikosta komento Add Module ja napsauta Open.

Uusi vakiomoduuli ilmestyy Code-ikkunaan.

7 Kirjoita vakiomoduuliin tällainen AddName-lausealiohjelma:

Tekstiruutuun lisätään uusi rivi merkkikoodeilla Chr(13) ja Chr(10).

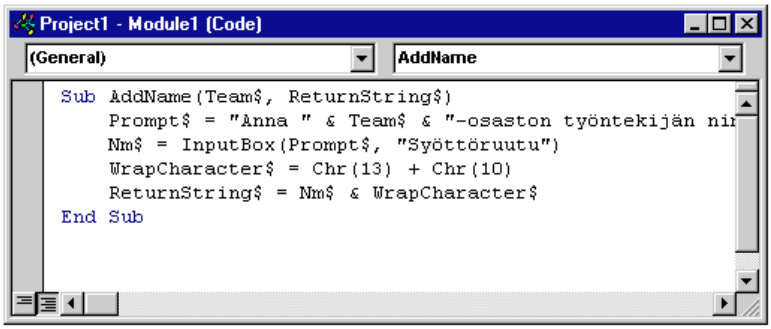
```
Sub AddName(Team$, ReturnString$)
    Prompt$ = "Anna " & Team$ & "-osaston  työntekijän  nimi."
    Nm$ = InputBox(Prompt$, "Syöttöruutu")
    WrapCharacter$ = Chr(13) + Chr(10)
    ReturnString$ = Nm$ & WrapCharacter$
End Sub
```

Tämä lausealiohjelma kysyy käyttäjältä työntekijän nimen InputBox-funktiolla. Sillä on kaksi argumenttia:

- Team$, osaston nimen sisältävä merkkijono
- ReturnString$, tyhjä merkkijonomuuttuja, joka palauttaa työntekijän muotoillun nimen kutsuvaan aliohjelmaan.

Ennen työntekijän nimen palauttamista lausealiohjelma lisää sen perään vaunupalautus- ja rivinvaihtomerkin, jotta tekstiruudun jokainen nimi tulee omalle rivilleen. Tämä on tavanomainen menettely, jota voit soveltaa mihin tahansa tekstiruutuun.

Code-ikkunan pitäisi näyttää tällaiselta:



8 Sulje Code-ikkuna ja kaksoisnapsauta lomakkeen ensimmäistä Lisää nimi -painiketta (Myynti-tekstiruudun alapuolella). Kirjoita tapahtuma-aliohjelmaan cmdSales_Click seuraavat lauseet:

```
AddName "Myynti", SalesPosition$
txtSales.Text = txtSales.Text & SalesPosition$
```

AddName-aliohjelman kutsussa on yksi arvoargumentti (”Myynti”) ja yksi viittausargumentti (SalesPosition$). Toinen lause käyttää viittausargumenttia lisätessään tekstin txtSales-tekstiruutuun. Yhteenliittämisoperaattori (&) lisää uuden nimen tekstiruudussa ennestään olevien nimien jälkeen.

9 Avaa Code-ikkunan objektiluettelo ja valitse cmdMkt-objekti. Kirjoita cmdMkt_Click-aliohjelmaan seuraavat lauseet:

```
AddName "Markkinointi", MktPosition$
txtMkt.Text = txtMkt.Text & MktPosition$
```

Tämä tapahtuma-aliohjelma on samanlainen kuin cmdSales_Click, mutta se lähettää AddName-aliohjelmaan merkkijonon ”Markkinointi” ja päivittää txtMkt-tekstiruudun sisällön. Lisäksi paikallisen palautusmuuttujan nimi on vaihdettu johdonmukaisemmaksi.

10 Avaa objektiluettelo ja valitse cmdQuit-objekti. Kirjoita cmdQuit_Click-aliohjelmaan **End** ja sulje Code-ikkuna.

11 Napsauta työkalurivin Tallenna projekti -painiketta. Valitse kansio \Vb5Sbs\Less09 ja tallenna vakiomoduuli nimellä **MyTeams.bas**. Tallenna lomake nimellä **MyTeams.frm** ja projekti nimellä **MyTeams.vbp**.

Tallenna projekti -painike

Siinä kaikki! Nyt voit käynnistää MyTeams-ohjelman.

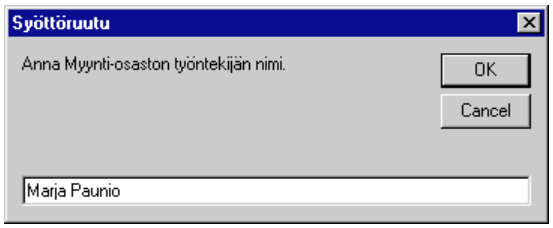
MyTeams-ohjelman suorittaminen

Käynnistä-painike

Valmis Teams-ohjelma on kansiossa \Vb5Sbs\Less09.

1. Käynnistä ohjelma napsauttamalla työkalurivin Käynnistä-painiketta.
2. Napsauta Myynti-tekstiruudun alla olevaa Lisää nimi -painiketta ja kirjoita Syöttöruutu-valintaikkunaan **Marja Paunio**.

   Valintaikkunan pitäisi näyttää tällaiselta:



3. Lisää nimi Myynti-tekstiruutuun napsauttamalla OK.

   Nimi ilmestyy Myynti-tekstiruutuun.
4. Napsauta Markkinointi-tekstiruudun alla olevaa Lisää nimi -painiketta, kirjoita valintaikkunaan **Henri Jaakkola** ja paina ENTER.

   Nimi ilmestyy Markkinointi-tekstiruutuun. Näytön pitäisi olla tällainen:

5. Kirjoita kumpaankin tekstiruutuun vielä muutama nimi. Tässä on tilaisuutesi ottaa töihin juuri haluamasi henkilöt!

   Kunkin nimen pitäisi olla tekstiruudussa omalla rivillään. Tekstiruudut eivät vierity automaattisesti, joten et näe kaikkia nimiä, jos niitä on enemmän kuin tekstiruutuun mahtuu. Voit tuoda haluamasi nimet esiin napsauttamalla vierityspalkkeja.
6. Kun olet valmis, lopeta ohjelma napsauttamalla Lopeta-painiketta.

Lisätietoja: Arvoargumentit

Lausealiohjelmia kuvatessani kerroin, että voit käyttää arvo- tai viittausargumentteja. Kun muuttuja välitetään viittausargumenttina (oletus), siihen tehdyt muutokset vaikuttavat kutsuvaan aliohjelmaan. Tätä käytettiin hyväksi MyTeams-ohjelmassa, kun arvoargumenttina välitetty nimi lisättiin tekstiruutuun. Viittausargumentit ovat todella hyödyllisiä, kunhan et vahingossa muuta niiden arvoja aliohjelmassa. Tutki esimerkiksi seuraavaa lausealiohjelman määrittelyä ja kutsua:

```
Sub CostPlusInterest(Cost, Total)
    Cost = Cost * 1.05  ' lisää kustannuksiin 5 %...
    Total = Int(Cost)   ' muuta tulos kokonaisluvuksi ja palaa
End Sub
.
.
.
Price = 100
Total = 0
CostPlusInterest Price, Total
Print Price; "on 5 %:n korolla "; Total
```

Varo viittaus-argumenteissa piileviä ansoja.

Tässä esimerkissä ohjelmoija välittää costPlusInterest-aliohjelmalle viittausargumentteina kaksi muuttujaa: Price ja Total. Hän aikoo käyttää päivitettyä Total-muuttujaa seuraavassa Print-metodissa, mutta hän on valitettavasti unohtanut, että Price-muuttuja on välillä päivittynyt. (Koska Price on viittausargumentti, Cost-argumenttiin tehdyt muutokset vaikuttavat siihen.) Tästä aiheutuu seuraava virheellinen tulos:

```
105 on 5 %:n korolla 105
```

ByVal-avainsana

Tällainen ongelma voidaan tietysti välttää siten, että aliohjelmalle välitetyn muuttujan arvoa ei koskaan muuteta. Tämä ratkaisu voi kuitenkin vaatia ylimääräisiä lauseita ja osoittautua epäluotettavaksi, jos ohjelmaa kehitellään ryhmätyönä. Parempi tapa on käyttää aliohjelman argumenttiluettelossa ByVal-avainsanaa. Tällöin Visual Basic säilyttää alkuperäisen argumentin kopion ja palauttaa sen muuttumattomana aliohjelman päättyessä — vaikka muuttujan arvo olisikin vaihdettu aliohjelman sisällä. ByVal-avainsanaa käytetään argumenttiluettelossa näin:

```
Sub CostPlusInterest(ByVal Cost, Total)
```

Kun Cost-argumentti esitellään käyttäen ByVal-avainsanaa, ohjelma tuottaa oikean tuloksen:

```
100 on 5 %:n korolla 105
```

Muuttujan välittäminen arvoargumenttina

Voit välittää muuttujan arvoargumenttina ympäröimällä sen sulkein.

Ellet halua turvautua ByVal-avainsanaan, voit käyttää vaihtoehtoista tapaa estää välitetyn muuttujan muuttaminen: muunna se vakioarvoksi ympäröimällä se sulkein. Tämä harvoin käytetty temppu toimii aina Visual Basicissa ja tekee aliohjelmien kutsuista intuitiivisempia. Jos varta vasten välität muuttujan arvona, jokainen huomaa, mitä tarkoitat. Muuttujan välittäminen arvona on *joskus* myös tehokkain ratkaisu. Voisit kutsua CostPlusInterest-aliohjelmaa ja välittää Price-muuttujan arvoargumenttina näin:

```
CostPlusInterest (Price), Total
```

Jos esimerkkiohjelmaa kutsutaan näin, se tuottaa oikean tuloksen:

```
100 on 5 %:n korolla 105
```

Tässä harjoituksessa opit käsittelemään tietoja käyttäen julkisia muuttujia, funktioaliohjelmia ja lausealiohjelmia. Käytä näitä rakenteita hyväksesi, kun ohjelmasi alkavat kasvaa. Huomaat kyllä, että ne säästävät paljon aikaa ja että voit käyttää ne uudelleen tulevissa projekteissa.

Jos haluat jatkaa seuraavaan harjoitukseen

➤ Jätä Visual Basic käyntiin ja siirry harjoitukseen 10.

Jos haluat lopettaa Visual Basicin

➤ Valitse File-valikosta komento Exit.

Jos näyttöön tulee Save-valintaikkuna, napsauta Yes.

Harjoituksen tiivistelmä

Tehtävä	Ohje	Painike
Antaa objektille nimi	Sijoita Name-ominaisuuteen yksikäsitteinen nimi. Noudata vakiintuneita nimeämissääntöjä, jotta tunnistat nimen.	
Lisätä uusi moduuli	Napsauta Lisää lomake -painikkeen alanuolta ja valitse luettelosta Module. *tai* Valitse Project-valikosta komento Add Module.	

Tehtävä	Ohje
Tallentaa uusi moduuli	Valitse moduuli Project-ikkunasta ja valitse File-valikosta komento Save Module1 As.
Poistaa moduuli ohjelmasta	Valitse moduuli Project-ikkunasta ja valitse Project-valikosta komento Remove.
Lisätä moduuli ohjelmaan	Valitse Project-valikosta komento Add File.
Luoda julkinen muuttuja	Esittele muuttuja vakiomoduulissa Public-avainsanalla. Esimerkki: `Public TotalSales  As Integer`

Luoda julkinen funktioaliohjelma	Kirjoita funktion lauseet vakiomoduuliin Function- ja End Function -avainsanojen väliin. Funktiot ovat normaalisti julkisia. Esimerkki:
	`Function Rate(hits, Attempts) As String`
	`    Percent = Hits / Attempts`
	`    Rate = Format(Percent, "0.0%")`
	`End Function`

Kutsua käyttäjän määrittämää funktioaliohjelmaa	Kirjoita funktion nimi ja tarvittavat argumentit aliohjelman ohjelmalauseeseen. Esimerkki: `lblRate.Caption = Rate(NumHits, NumTrys)`

Luoda lause-aliohjelma	Kirjoita lausealiohjelman vakiomoduuliin Sub- ja End Sub -avainsanojen väliin. Lausealiohjelmat ovat normaalisti julkisia. Esimerkki:
	`Sub CostPlusInterest(Cost, Total)`
	`    Cost = Cost * 1.05`
	`    Total = Int(Cost)`
	`End Sub`

Kutsua lause-aliohjelmaa	Kirjoita lausealiohjelman nimi ja tarvittavat parametrit aliohjelman ohjelmalauseeseen. Esimerkki: `CostPlusInterest PriceTag, TotalPrice`
Määrittää tai käyttää objektin ominaisuutta yleiskäyttöisessä aliohjelmassa	Kirjoita lomakkeen nimi, piste (.) ja objektin nimi. Esimerkki: `Form1.Label1.Caption`

Tehtävä	Ohje

Välittää arvo-argumentti	Kirjoita aliohjelman argumentiksi sulkein ympäröity muuttujan nimi tai vakioarvo. Esimerkki:
	`CalculateInterest (Price)`
	tai
	`CalculateInterest 500`

Välittää viittaus-argumentti	Käytä aliohjelman argumenttina muuttujaa: `CalculateInterest Price`

Kun tarvitset tietoja aiheesta	Valitse Visual Basicin Help-valikosta Microsoft Visual Basic Help Topics, valitse Index-välilehti ja...
Objektien nimeäminen	Kirjoita **Naming**.
Julkisten muuttujien luominen	Kirjoita **Public statement**.
Funktioaliohjelmien määrittely	Kirjoita **Function procedures**.
Lausealiohjelmien määrittely	Kirjoita **Sub**.

Tietoja seuraavasta harjoituksesta

Seuraavassa harjoituksessa opit käsittelemään objekteja ja muuttujia ryhminä, joita sanotaan *kokoelmiksi* ja *taulukoiksi*. Näet, miten ohjelmassa käytetään Controls-kokoelmaa, miten kokoelmia käsitellään For...Each-silmukalla ja miten suuria muuttujajoukkoja tallennetaan ja käytetään yksi- ja kaksiulotteisten taulukoiden avulla.

Harjoitus

10

Kokoelmien ja taulukoiden käsittely

Arvioitu aika
45 min.

Tässä harjoituksessa opit

- käyttämään kokoelmia
- käsittelemään kokoelmia For Each…Next -silmukassa
- järjestämään muuttujia taulukoiksi.

Lomakkeen objekteja voidaan sijoittaa ryhmiin, joita sanotaan *kokoelmiksi*. Tässä harjoituksessa tutustut Visual Basic -ohjelman vakiokokoelmiin ja näet, miten kokoelmia käsitellään erikoissilmukassa nimeltä For Each...Next. Opit myös järjestämään muuttujia *taulukoiksi*. Taulukot helpottavat tietojen hallitsemista ja ovat hyvä johdanto tietokantaohjelmoinnin tekniikoihin, joita pääset kokeilemaan harjoituksessa 11.

Objektikokoelmien käsittely

Kokoelma on toisiinsa liittyvien objektien ryhmä.

Tiedät jo, että lomakkeen objektit tallennetaan samaan tiedostoon. Tiesitkö senkin, että Visual Basic tulkitsee lomakkeen objektit saman ryhmän jäseniksi? Lomakkeen kaikkien objektien ryhmää sanotaan Visual Basicissa *Controls-kokoelmaksi*. Se syntyy automaattisesti, kun avaat uuden lomakkeen, ja laajenee, kun lisäät lomakkeeseen objekteja. Itse asiassa Visual Basic ylläpitää useita objektien vakiokokoelmia, joita voit käyttää ohjelmia kirjoittaessasi. Tämän harjoituksen ensimmäisessä osassa opit perustaidot, jotka riittävät minkä tahansa kokoelman hallitsemiseen.

Kokoelmien merkintätapoja

Ohjelman jokaisella kokoelmalla on oma nimensä, jotta siihen voidaan viitata erillisenä yksikkönä. Lomakkeen kaikki objektit sisältävä kokoelma on siis nimeltään Controls. Koska ohjelmassa voi olla useita lomakkeita (ja siten useita Controls-kokoelmia), Controls-kokoelman nimen eteen pitää lisätä lomakkeen nimi, jos lomakkeita on enemmän kuin yksi. Esimerkiksi Form1-lomakkeen Controls-kokoelmaan viitataan tällaisella nimellä:

```
Form1.Controls
```

Jokaisella lomakkeella on Controls-kokoelma.

Objektin nimen ja Controls-avainsanan välinen piste saa Controls-kokoelman näyttämään ominaisuudelta, mutta Visual Basic -ohjelmoijat kuvaavat Controls-kokoelman objektiksi, jonka Form1-objekti sisältää. Tällainen objektien suhde tai hierarkia muistuttaa hieman polkunimen kansioiden hierarkiaa; näet tämän merkintätavan uudelleen alkaessasi käyttää sovellusobjekteja harjoituksessa 12.

Voit paitsi muokata omien ohjelmiesi objekteja ja kokoelmia, myös etsiä järjestelmästä muita sovellusobjekteja ja käyttää niitä ohjelmissasi. Palaamme tähän aiheeseen harjoituksessa 12, jossa kuvataan Visual Basic Object Browseria.

Viittaaminen kokoelman objekteihin

Voit viitata kokoelman objekteihin eli erillisiin jäseniin usealla tavalla. Ensimmäinen tapa on määrittää objekti kirjoittamalla sen nimi sijoituslauseeseen. Esimerkiksi lause

```
Form1.Controls!Label1.Caption = "Työntekijät"
```

valitsee Controls-kokoelmasta Label1-objektin ja sijoittaa sen Caption-ominaisuuteen tekstin ”Työntekijät”. Huutomerkki (!) on linkki Label1-objektin ja Controls-kokoelman välillä. Vaikka tämä lause saattaa näyttää pelkältä suupalalta Visual Basic -kääntäjälle, se kuvaa kokoelman sisäistä hierarkiaa täsmällisesti.

Toinen tapa viitata kokoelman objektiin on määrittää sen *indeksinumero*. Näin voit viitata tiettyyn objektiin tai käydä kokoelman objektit läpi silmukassa. Visual Basic säilyttää kokoelmaobjektit käänteisessä luontijärjestyksessä, joten voit viitata esimerkiksi lomakkeen viimeksi luotuun objektiin nollaindeksillä näin:

Voit viitata kokoelman objekteihin yksitellen tai ryhminä.

```
Form1.Controls(0).Caption = "Yhtiö"
```

Tämä lause määrittää lomakkeen viimeksi luodun objektin Caption-ominaisuudeksi ”Yhtiö”. (Toiseksi viimeinen objekti on indeksiltään 1, kolmanneksi viimeinen 2 ja niin edelleen.)

For Each…Next -silmukoiden kirjoittaminen

Vaikka voit viitata kokoelman jäseniin yksitellen, kokoelman objekteja on parasta käsitellä ryhmänä. Kokoelmathan on luotu juuri sitä varten, että objektiryhmiä voitaisiin käsitellä tehokkaasti. Voit esimerkiksi näyttää, siirtää, lajitella, nimetä, suurentaa tai pienentää kokoelman kaikki objektit kerralla.

For Each…Next -silmukat on tarkoitettu kokoelmien käsittelemiseen.

Tällaisiin tehtäviin sopii erikoissilmukka nimeltä For Each…Next. Se käy kokoelman objektit läpi yksitellen. For Each…Next -silmukka on samankaltainen kuin For…Next -silmukka, johon tutustuit harjoituksessa 6. Controls-kokoelman kanssa käytettynä For Each…Next -silmukka näyttää tällaiselta:

```
For Each Objekti in Lomake.Controls
    objektin käsittely
Next Control
```

Objekti on erikoismuuttuja, joka edustaa nykyistä objektia kokoelmassa, ja *Lomake* on lomakkeen nimi. Silmukan rungossa on kokoelman objekteihin kohdistuvia lauseita. Voisit esimerkiksi muuttaa kokoelman objektien Enabled-, Left-, Top-, Caption- tai Visible-ominaisuutta tai näyttää objektien nimet luetteloruudussa.

Control-muuttuja edustaa nykyistä objektia For Each…Next -silmukassa.

Objektikokoelman siirtäminen

Seuraavassa ohjelmassa käytät Controls-kokoelmaa, kun siirrät joukon objekteja samanaikaisesti lomakkeen vasemmasta reunasta oikeaan. Ohjelma suorittaa objekteja siirtävän For Each…Next -silmukan joka kerta, kun käyttäjä napsauttaa MoveObjects -komentopainiketta. Joskus kokoelman eräät objektit edellyttävät erikoiskäsittelyä, ja siksi tämän tehtävän jälkeen muutat ohjelmaa niin, että se siirtää muita objekteja paitsi komentopainiketta.

Controls-kokoelman käsittely For Each…Next -silmukalla

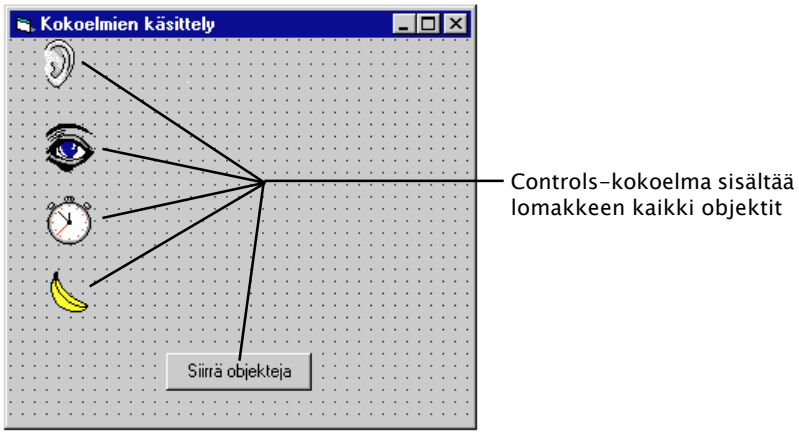
1. Käynnistä Visual Basic.

   New Project -valintaikkuna avautuu.

2. Napsauta Existing-välilehteä ja avaa kansiosta \Vb5Sbs\Less10 projekti Move.vbp.

3. Valitse Project-ikkunasta Move.frm ja napsauta Näytä objekti -painiketta.

   Ohjelmointiympäristössä näkyy Kokoelmien käsittely -lomake, joka on esitetty seuraavalla sivulla.



Lomakkeessa on viisi objektia, jotka kuuluvat Controls-kokoelmaan. Objektien Picture-, Name- ja Caption-ominaisuudet on jo määritetty, mutta sinun pitää lisätä lauseet, jotka siirtävät kokoelman lomakkeen poikki.

4 Kaksoisnapsauta lomakkeen Siirrä objekteja -painiketta.

Tapahtuma-aliohjelma cmdButton_Click ilmestyy Code-ikkunaan.

5 Kirjoita seuraavat ohjelmalauseet:

```
For Each Ctrl In Controls
    Ctrl.Left = Ctrl.Left + 200
Next Ctrl
```

Aina käyttäjän napsautettua Siirrä objekteja -painiketta tämä For Each... Next-silmukka käy läpi Controls-kokoelman objektit ja siirtää kutakin objektia 200 twip-yksikköä oikealle. (Voisit siirtää objekteja vasemmalle vähentämällä Left-ominaisuudesta 200.) Ctrl-muuttuja edustaa kokoelman nykyistä objektia ja sisältää samat ominaisuusasetukset kuin nykyinen objekti. Tässä silmukassa muutetaan Left-ominaisuutta, joka määrittää objektin sijainnin suhteessa lomakkeen vasempaan reunaan.

Kun jokaisen objektin Left-ominaisuutta muutetaan For Each...Next -silmukassa, objektit siirtyvät ryhmänä.

6 Valitse File-valikosta komento Save Move.frm As ja tallenna lomake nimellä **MyMove.frm**.

7 Valitse File-valikosta komento Save Project As ja tallenna projekti nimellä **MyMove.vbp**.

8 Sulje Code-ikkuna ja napsauta työkalurivin Käynnistä-painiketta.

Käynnistä-painike

Ohjelma käynnistyy. Sen lomakkeen vasemmassa laidassa näkyy neljä kuvaketta ja alalaidassa komentopainike.

Myös Siirrä objekteja -painike siirtyy napsautettuasi sitä.

9 Napsauta Siirrä objekteja -painiketta useita kertoja.

Joka kerta painiketta napsautettuasi lomakkeen objektit siirtyvät hieman oikealle. Komentopainike siirtyy kuvien mukana, koska sekin kuuluu Controls-kokoelmaan.

10 Lopeta ohjelma napsauttamalla työkalurivin Lopeta-painiketta.

Lopeta-painike

Kaikkia objekteja ei ole pakko siirtää samanaikaisesti. Visual Basicissa on mahdollista muokata kokoelman jäseniä tarvittaessa erikseen. Seuraavassa tehtävässä opit pitämään Move Objects -painikkeen paikallaan, kun kuvat siirtyvät oikealle.

Tag-ominaisuuden käyttö For Each...Next -silmukassa

Jos haluat käsitellä kokoelman yhtä tai useaa jäsentä eri tavoin kuin muita, käytä Tag-ominaisuutta. Määritä, millaisen Tag-ominaisuuden sisältävät objektit otetaan huomioon. Ohjelma lukee kunkin objektin Tag-ominaisuuden käsitellessään kokoelmaa For Each...Next -silmukassa ja päättää sen perusteella, muokataanko objektia tavallisella tavalla vai vaatiiko se erikoistoimia.

Tag-ominaisuudella merkitään objektit, jotka on käsiteltävä silmukassa poikkeavalla tavalla.

Oletetaan, että olet sijoittanut MyMove-ohjelman imgBanana-objektin Tag-ominaisuuteen sanan *Slow*. Voisit kirjoittaa If...Then-lauseen, joka havaitsee Slow-asetuksen silmukassa ja siirtää banaania lyhyemmän matkan kuin muita objekteja.

VIHJE Jos useita objekteja pitää muokata erikoisella tavalla For Each...Next -silmukassa, käytä Select Case -rakennetta tai If...Then-rakenteessa ElseIf-lauseita.

Seuraavassa ohjelmassa annetaan cmdButton-objektille Tag-ominaisuudeksi ”Button”, jotta objektin siirtäminen For Each...Next -silmukassa voidaan estää.

Erikoistoimien valitseminen ominaisuuden perusteella

1 Valitse File-valikosta komento Save MyMove.frm As ja tallenna lomake nimellä **MyTag.frm**.

Tallenna ohjelma uudella nimellä ennen kuin teet muutoksia, jotta alkuperäinen MyMove-projekti säilyy ennallaan.

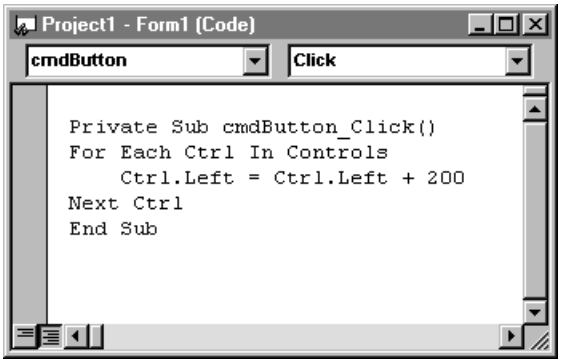
2 Valitse File-valikosta komento Save Project As ja tallenna projekti nimellä **MyTag.vbp**.

3 Napsauta lomakkeen Siirrä objekteja -painiketta ja avaa Properties-ikkuna.

4 Määritä cmdButton-objektin Tag-ominaisuudeksi ”Button”.

5 Kaksoisnapsauta lomakkeen Siirrä objekteja -painiketta.

Tapahtuma-aliohjelma cmdButton_Click ilmestyy Code-ikkunaan seuraavan kuvan kaltaisena:



6 Muokkaa tapahtuma-aliohjelma alla esitettyyn muotoon. Kolmas ja viides lause ovat uusia, ja neljättä lausetta on sisennetty hieman lisää.

```
Private Sub cmdButton_Click()
For Each Ctrl In Controls
    If Ctrl.Tag <> "Button" Then
        Ctrl.Left = Ctrl.Left + 200
    End If
Next Ctrl
End Sub
```

If...Then-lause tutkii, onko Tag-ominaisuus ”Button”.

Tämän For Each...Next -silmukan uusi piirre on If...Then-lause, joka tutkii, onko kokoelman jäsenen Tag-ominaisuus ”Button”. Jos on, kyseinen objekti ohitetaan siirtämättä sitä mihinkään. Merkkijonolla ”Button” ei ole erikoismerkitystä Visual Basicissa — se on vain sana, jonka valitsin komentopainikeobjektin tunnukseksi. Olisin yhtä hyvin voinut kirjoittaa Tag-ominaisuuteen ”Ei siirry” tai “Ohita”.

7 Sulje Code-ikkuna ja napsauta työkalurivin Käynnistä-painiketta.

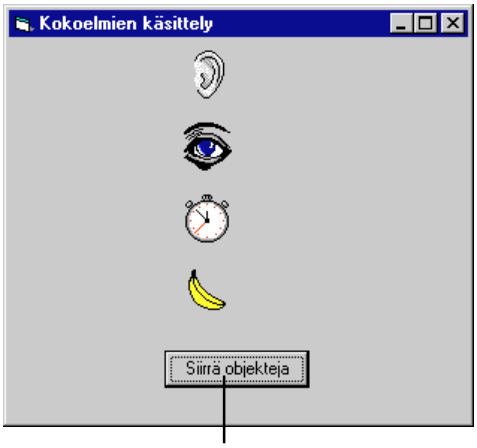
Käynnistä-painike

Ohjelma käynnistyy ja näyttää käyttöliittymän viisi objektia lomakkeessa.

8 Napsauta Siirrä objekteja -painiketta seitsemän tai kahdeksan kertaa.

Painiketta napsautettuasi lomakkeen kuvat siirtyvät oikealle lomakkeen poikki. Komentopainike sen sijaan pysyy paikallaan:

Valmis Tag-ohjelma on kansiossa \Vb5Sbs\Less10.



Tag-ominaisuus on keino käsitellä tietyt objektit erikoisella tavalla

Kokoelman yhden objektin käsitteleminen muista poikkeavalla tavalla voi olla hyvin tärkeää. Tässä tapauksessa Tag-ominaisuuden käyttäminen For Each…Next -silmukassa paransi ohjelman käyttöliittymää huomattavasti. Kun käytät muunlaisia Visual Basicin kokoelmia, pidä Tag-ominaisuus mielessäsi.

Lopeta-painike

9 Lopeta ohjelma napsauttamalla työkalurivin Lopeta-painiketta.

10 Napsauta työkalurivin Tallenna projekti -painiketta, jotta tekemäsi muutokset tallentuvat tiedostoon MyTag.vbp.

Muita Visual Basicin kokoelmia

Visual Basicissa on Controls-kokoelman lisäksi seuraavat valmiit kokoelmat, joita voit käyttää ohjelmissasi. Jos haluat lisätietoja näistä, etsi Visual Basicin ohjeista hakusanaa *collections*.

Kokoelma	Kuvaus
Forms-kokoelma	Kaikkien ohjelmaan ladattujen lomakkeiden kokoelma. For Each…Next -silmukassa voit asettaa yhden tai usean lomakkeen tai niiden sisältämien objektien ominaisuuksia.
Printers-kokoelma	Kaikkien järjestelmään asennettujen tulostimien kokoelma. Käyttämällä For Each…Next -silmukkaa ja AddItem-metodia voit näyttää kaikki käytettävissä olevat tulostimet luetteloruudussa ja antaa käyttäjän valita niistä yhden.
Tietokantakokoelmat	Joukko kokoelmia,jotka liittyvät tietojen hakemiseen ja tietokannan hallintaan. Erityisen hyödyllisiä tietokantakokoelmia ovat Columns, Containers, Indexes ja Databases. Opit näistä enemmän seuraavassa harjoituksessa.
Sovellusobjekti-kokoelmat	Microsoft Office 97:ssä kaikkien sovellusten komennot ja toiminnot on järjestetty kokoelmiksi. Esimerkiksi Word 97:n Documents-kokoelma sisältää kaikki asiakirjat, jotka ovat avoinna Microsoft Wordissa.

Muuttujataulukoiden käsittely

Harjoituksessa 6 teit leikkaamalla ja liittämällä objektitaulukon, johon tallensit useita kuvakehysobjekteja samalla nimellä. Koska jokaisella taulukon objektilla oli sama nimi, pystyit käsittelemään kuvakehysten joukon yhdellä For...Next-silmukalla.

Taulukko on yhdellä nimellä tallennettujen arvojen kokoelma.

Seuraavaksi opit tallentamaan muuttujia taulukkoon samankaltaisilla menetelmillä. Muuttujataulukot (lyhyemmin *taulukot*) ovat objektitaulukoiden ja kokoelmien tapaan keino, jolla viitataan kokonaiseen arvoryhmään yhdellä nimellä ja muokataan arvoja yksitellen tai ryhmänä For...Next- tai Do-silmukassa.

Taulukot ovat käteviä, koska niillä on helppo pitää kirjaa suuristakin tietomääristä. Perinteiset muuttujat eivät oikein hyvin sovellu tällaisiin tehtäviin. Kuvittele esimerkiksi yhdeksästä vaihtovuorosta muodostuvan baseball-pelin tulostaulua. Arvelet ehkä, että tulosten tallentamista varten tarvitaan kaksi yhdeksän muuttujan ryhmää (yhteensä 18 muuttujaa). Saattaisit antaa niille vaikkapa tällaiset nimet: Sisavuoro1Koti, Sisavuoro1Vieras ja niin edelleen.

Muuttujien käsitteleminen yksitellen vaatisi melkoisesti aikaa ja ohjelmalauseita. Onneksi Visual Basicissa on mahdollista järjestää tällaiset muuttujaryhmät taulukkoon, jolla on yksi yhteinen nimi ja helppokäyttöiset indeksit. Voisit esimerkiksi tehdä kaksiulotteisen (2 x 9) taulukon nimeltä Tulostaulu ja tallentaa siihen pelin tulokset. Katsotaan, miten tämä menetelmä toimii.

Taulukon luominen

Taulukko on esiteltävä, jotta sitä voitaisiin käyttää.

Taulukot luodaan eli *esitellään* samalla tavoin kuin muuttujat, ja esittelyn sijainti määrää, missä taulukkoa voidaan käyttää eli mikä on sen *näkyvyysalue*. Jos taulukko esitellään paikallisesti, siihen voidaan viitata vain siinä aliohjelmassa, jossa se on esitelty. Jos taulukko esitellään julkisesti vakiomoduulissa, siihen voidaan viitata missä tahansa. Taulukon esittelylauseeseen pitää kirjoittaa seuraavan taulukon mukaiset tiedot.

Taulukon esittelylauseen osa	Kuvaus
Taulukon nimi	Nimi, joka edustaa taulukkoa ohjelmassa. Taulukot nimetään yleensä samojen sääntöjen mukaan kuin muuttujat. (Lisätietoja muuttujista on harjoituksessa 4.)
Tietotyyppi	Taulukkoon tallennettavien tietojen tyyppi. Taulukon kaikki muuttujat ovat tavallisesti samaa tyyppiä. Jos et ole varma, millaisia tietoja taulukkoon on tarkoitus tallentaa tai jos tietojen tyyppi vaihtelee, valitse Variant- eli muunnelmatyyppi. Muuten voit valita jonkin perustietotyypeistä.
Ulotteisuus	Taulukon ulottuvuuksien määrä. Useimmat taulukot ovat yksiulotteisia (listoja) tai kaksiulotteisia (rivejä ja sarakkeita), mutta voit lisätä ulottuvuuksien määrää, jos kyseessä on monimutkainen matemaattinen malli, kuten kolmiulotteinen muoto.
Alkioiden määrä	Taulukkoon tallennettavien tietoalkioiden määrä. Alkioihin viitataan indeksinumerolla. Oletuksen mukaan taulukon ensimmäinen indeksi on 0 (nolla), samoin kuin objektitaulukoissa.

HUOMAA Taulukot, joihin voidaan tallentaa vain tietty määrä alkioita, ovat *vakiokokoisia*. Taulukot, joiden alkioiden määrä vaihtelee eli jotka voivat laajentua ohjelman suorituksen aikana, ovat *dynaamisia*.

Vakiokokoisen taulukon esitteleminen

Vakiokokoisen julkisen taulukon esittelylause näyttää tällaiselta:

```
Public Taulukko(Ulott1Alkiot, Ulott2Alkiot, ...) As Tietotyyppi
```

Seuraavat argumentit ovat tärkeitä:

Public-avainsanalla luodaan julkinen eli yleinen taulukko.

- Public on avainsana, joka luo yleisen eli julkisen taulukon.
- *Taulukko* on taulukon nimi.
- *Ulott1Alkiot* on taulukon ensimmäisen ulottuvuuden alkioiden määrä.
- *Ulott2Alkiot* on taulukon toisen ulottuvuuden alkioiden määrä (ulottuvuuksia voi olla enemmänkin).
- *Tietotyyppi* on avainsana, joka ilmaisee taulukkoon tallennettavien tietojen tyypin.

Koska taulukko on julkinen, sen esittely pitää sijoittaa projektin vakiomoduuliin (muiden julkisten eli yleisten muuttujien joukkoon).

HUOMAA Kun haluat esitellä taulukon paikallisesti tapahtuma-aliohjelmassa, korvaa Public-avainsana Static-avainsanalla ja sijoita esittely tapahtuma-aliohjelmaan. Paikallista taulukkoa voidaan käyttää vain siinä aliohjelmassa, jossa se on esitelty.

Jos esimerkiksi tarvitset julkisen ja yksiulotteisen Employees-nimisen merkkijonotaulukon, jossa on tilaa 10 työntekijän nimelle, kirjoita vakiomoduuliin tällainen lause:

```
Public Employees(9) As String
```

Taulukon ensimmäisen alkion indeksi on normaalisti 0.

Kun luot taulukon, Visual Basic varaa sille muistitilaa. Seuraavan sivun ylempi kuva esittää taulukon rakennetta. Taulukon 10 alkiota on numeroitu 0–9 eikä 1–10, koska taulukoiden indeksit alkavat nollasta, ellei Option Base -lausetta ole käytetty. Lisätietoja on seuraavan aukeaman kohdassa ”Option Base -lause”.

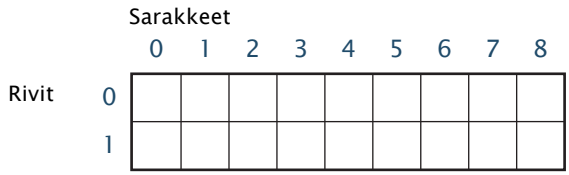
Jos haluat esitellä julkisen ja kaksiulotteisen Scoreboard-nimisen taulukon, jossa on kaksi riviä ja yhdeksän saraketta muunnelmatyyppisille tiedoille, kirjoita vakiomoduuliin tällainen lause (vastaavanlaista taulukkoa tarvitaan myöhemmin tässä harjoituksessa):

```
Public Scoreboard(1, 8) As Variant
```

Kaksiulotteisessa taulukossa tarvitaan kaksi indeksiä.

Visual Basic varaa taulukolle muistitilaa. Sitten voit käyttää taulukkoa ohjelmassa aivan kuin se olisi seuraavan kuvan kaltainen ruudukko. Tässä tapauksessa alkioiden rivi-indeksi on 0 tai 1 ja sarakeindeksi 0–8.

Scoreboard



Taulukon alkioiden käsittely

Kun olet esitellyt taulukon Static- tai Public-avainsanalla, voit käyttää sitä ohjelmassa. Viittaa sen alkioon kirjoittamalla taulukon nimi ja sulkein ympäröity indeksi. Indeksin on oltava kokonaisluku, esimerkiksi vakioarvo tai kokonaislukumuuttuja. (Indeksi on usein For...Next-silmukan laskurimuuttuja.)

Seuraava lause sijoittaisi edellä esitetyn Employees-taulukon alkioon 5 merkkijonon ”Leslie”:

```
Employees(5) = "Leslie"
```

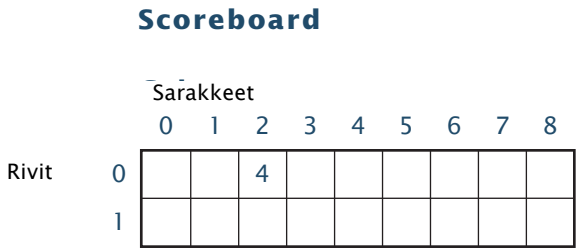
Lauseen suorittamisen jälkeen Employees-taulukon sisältö näyttäisi tällaiselta:

Taulukot säilyvät ohjelman suorituksen ajan järjestelmä- eli RAM-muistissa.

Vastaavasti seuraava lause sijoittaisi Scoreboard-taulukon rivin 0 sarakkeeseen 2 numeron 4:

```
Scoreboard(0, 2) = 4
```

Scoreboard-taulukko näyttää nyt tällaiselta:



Näillä indeksointimenetelmillä voit määrittää tai lukea taulukon minkä tahansa alkion arvon.

Option Base -lause

Jos mielestäsi olisi selkeämpää, että taulukon ensimmäisen alkion numero olisi 1 eikä 0, voit kirjoittaa vakiomoduuliin tällaisen Option Base -lauseen:

```
Option Base 1
```

Lause määrää, että ohjelman jokaisen taulukon indeksointi alkaa ykkösestä. Seuraavaksi kuvattavassa ohjelmassa käytetään Option Base -lausetta juuri tällä tavoin.

Vakiokokoisen lämpötilataulukon luominen

FixArray-ohjelma tallentaa viikon kunkin päivän ylimmän lämpötilan taulukkoon.

Seuraavassa ohjelmassa käytetään yksiulotteista julkista taulukkoa nimeltä Temperatures, johon tallennetaan viikon jokaisen päivän ylin lämpötila. Ohjelma kuvaa, miten taulukon avulla säilytetään ja käsitellään joukko toisiinsa liittyviä arvoja. Lämpötilat sijoitetaan taulukkoon InputBox-funktiolla, jota kutsutaan For...Next-silmukassa. Silmukkalaskurilla viitataan taulukon kuhunkin alkioon. Sitten taulukon sisältö näytetään lomakkeessa käyttäen For...Next-silmukkaa ja Print-metodia. Ohjelma myös laskee ja näyttää keskimääräisen lämpötilan.

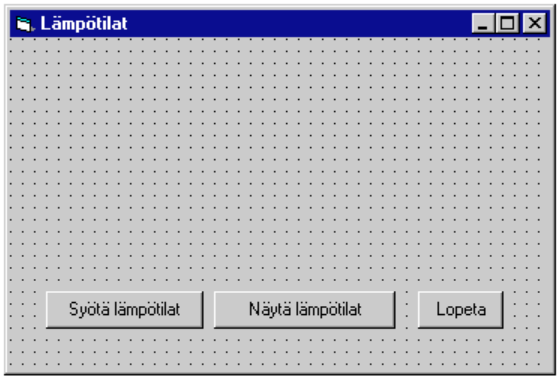
Vakiokokoisen taulukon käyttö

1. Valitse File-valikosta komento New Project ja napsauta OK.
2. Napsauta Komentopainike-työkalua ja lisää lomakkeen alareunaan kolme komentopainiketta.
3. Määritä komentopainikeobjektien ja lomakeobjektin ominaisuudet näin:

Objekti	Ominaisuus	Asetus
Command1	Caption	”Syötä lämpötilat”
	Name	cmdEnterTemps
Command2	Caption	”Näytä lämpötilat”
	Name	cmdDisplayTemps
Command3	Caption	”Lopeta”
	Name	cmdQuit
Form1	Caption	”Lämpötilat”
	AutoRedraw	True

TÄRKEÄÄ Määritä lomakkeen AutoRedraw-ominaisuudeksi True aina, kun näytät tietoja Print-metodilla. Tällöin Visual Basic päivittää näytön automaattisesti, jos lomake jää toisen ikkunan taakse.

4 Lomakkeen pitäisi näyttää tällaiselta:



5 Valitse Project-valikosta komento Add Module ja napsauta Open, jotta Visual Basic luo taulukon esittelyä varten vakiomoduulin.

Vakiomoduuli ilmestyy Code-ikkunaan.

6 Kirjoita vakiomoduuliin seuraavat lauseet:

```
Option Base 1
Public Temperatures(7) As Variant
```

Option Base -lause määrittää taulukon aloitusindeksiksi 1.

Option Base -lause vaihtaa ohjelman jokaisen taulukon ensimmäisen alkion indeksin nollasta ykköseksi. Toinen lause luo julkisen Variant- eli muunnelmatyyppisen Temperatures-nimisen taulukon, jossa on seitsemän alkiota. Koska taulukko on esitelty julkiseksi, se on käytettävissä kaikkialla ohjelmassa.

7 Sulje vakiomoduulin Code-ikkuna ja kaksoisnapsauta Syötä lämpötilat -painiketta.

Tapahtuma-aliohjelma cmdEnterTemps_Click ilmestyy Code-ikkunaan.

8 Kirjoita seuraavat ohjelmalauseet, jotka kysyvät käyttäjältä lämpötilat ja tallentavat ne taulukkoon:

```
Cls
Prompt$ = "Anna  päivän  ylin  lämpötila."
For i% = 1 To 7
    Title$ = "Päivä " & i%
    Temperatures(i%) = InputBox(Prompt$, Title$)
Next i%
```

HUOMAA Tapahtuma-aliohjelman alussa oleva Cls-metodin kutsu poistaa lomakkeesta tiedot, jotka siihen on ehkä lisätty aikaisemmin. Näin voit syöttää ohjelmalla useampia kuin yhden lämpötilasarjan.

Laskurimuuttuja i% toimii taulukon indeksinä.

For…Next-silmukka käyttää silmukkalaskuria i% taulukon indeksinä tallentaessaan lämpötila-arvot taulukon alkioihin 1–7. Syöttötiedot lukee InputBox-funktio, jossa on argumentteina muuttujat Prompt$ ja Title$.

9 Avaa Code-ikkunan objektiluettelo ja valitse cmdDisplayTemps-objekti. Kirjoita tapahtuma-aliohjelmaan cmdDisplayTemps_Click seuraavat lauseet:

```
Print "Viikon  ylimmät  lämpötilat:"
Print
For i% = 1 To 7
    Print "Päivä "; i%, Temperatures(i%)
    Total! = Total! + Temperatures(i%)
Next i%
Print
Print "Ylimpien  lämpötilojen  keskiarvo:  "; Total! / 7
```

Tämä aliohjelma näyttää Temperatures-taulukkoon tallennetut tiedot Print-metodilla. Se käy läpi taulukon alkiot For…Next-silmukassa ja laskee lämpötilojen summan lauseella

```
Total! = Total! + Temperatures(i%)
```

Aliohjelman viimeinen lause näyttää viikon keskilämpötilan eli lämpötilojen summan jaettuna päivien määrällä.

Valmis FixArray-ohjelma on kansiossa \Vb5Sbs\Less10.

10 Avaa Code-ikkunan objektiluettelo ja napsauta cmdQuit-objektia. Kirjoita tapahtuma-aliohjelmaan cmdQuit_Click tämä lyhyt lause:

```
End
```

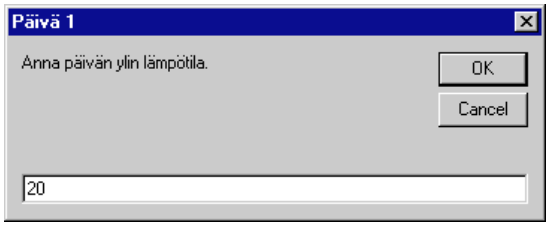
11 Napsauta työkalurivin Tallenna projekti -painiketta ja tallenna vakiomoduuli, lomake ja projekti. Valitse kansio \Vb5Sbs\Less10 ja tallenna kukin tiedosto nimellä **MyFixArray**.

Tallenna projekti -painike

12 Käynnistä ohjelma napsauttamalla Käynnistä-painiketta.

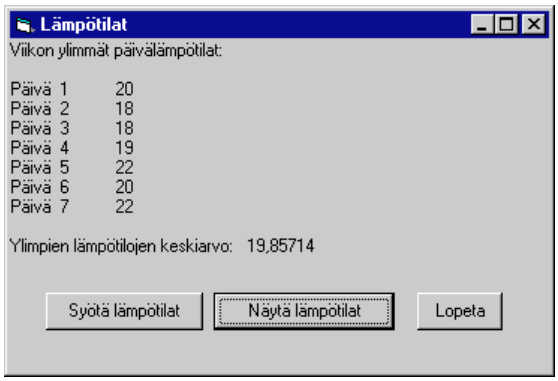
13 Napsauta Syötä lämpötilat -painiketta ja kirjoita seitsemän lämpötila-arvoa, kun InputBox-funktio kysyy niitä.

InputBox-funktion valintaikkunan pitäisi näyttää suunnilleen tällaiselta:



14 Kun olet syöttänyt lämpötilat, napsauta Näytä lämpötilat -painiketta.

Ohjelma luettelee lämpötilat lomakkeessa Print-metodilla ja näyttää lopuksi keskilämpötilan. Lomake näyttää tällaiselta:



15 Lopeta ohjelma napsauttamalla Lopeta-painiketta.

Dynaamisen taulukon luominen

Kuten huomaat, taulukot ovat hyvin näppärä tapa käsitellä numeroluetteloita, varsinkin jos samalla käytetään For...Next-silmukoita. Entä jos et tiedä ennen ohjelman alkamista, kuinka paljon taulukkotilaa tarvitaan? Entä jos käyttäjän pitää pystyä valitsemaan, kuinka monta lämpötila-arvoa hän syöttää MyFixArray-ohjelmalla?

Visual Basicissa tämä ongelma on ratkaistu tehokkaasti muodostamalla joustava tietosäilö, jota sanotaan *dynaamiseksi taulukoksi*. Dynaamisen taulukon koko määritetään ohjelman suorituksen aikana joko käyttäjän tekemän valinnan tai ohjelmassa olevien ehtojen perusteella. Dynaamisen taulukon koon määrittämiseen kuuluu useita vaiheita. Vaikka taulukon todellinen koko selviää vasta ohjelmaa suoritettaessa, sinun pitää tehdä taulukkoa varten ”varauksia” jo

Dynaamisten taulukoiden koko määritetään suorituksen aikana.

suunnitteluvaiheessa. Dynaamisen taulukon luomiseen kuuluu seuraavat perusvaiheet:

1. Määritä taulukon nimi ja tyyppi ohjelmaa kirjoittaessasi, mutta jätä alkioiden määrä avoimeksi. Voit esimerkiksi luoda julkisen dynaamisen taulukon nimeltä Temperatures tällaisella lauseella:

   ```
   Public Temperatures() as Variant
   ```

2. Lisää lauseet, joilla saadaan selville taulukon alkioiden määrä ohjelmaa suoritettaessa. Voit kysyä taulukon koon käyttäjältä InputBox-funktiolla tai laskea sen objektien ominaisuuksien tai muiden ehtojen perusteella. Esimerkiksi seuraava lause kysyy taulukon koon käyttäjältä ja sijoittaa sen Days-muuttujaan:

   ```
   Days = InputBox("Kuinka  monta  päivää?", "Luo  taulukko")
   ```

3. Käytä muuttujaa ReDim-lauseessa, joka määrittää taulukon koon. Esimerkiksi seuraava lause valitsee Temperatures-taulukon koon Days-muuttujan mukaan:

   ```
   ReDim Temperatures(Days)
   ```

4. Käytä muuttujaa ylärajana For…Next-silmukassa, jolla käydään läpi taulukon alkiot. Esimerkiksi seuraavassa silmukassa ylärajana on Days-muuttuja:

   ```
   For i% = 1 to Days
       Temperatures(i%) = InputBox(Prompt$, Title$)
   Next i%
   ```

Seuraavassa tehtävässä muokkaat MyFixArray-ohjelmaa näiden vaiheiden mukaisesti niin, että se pystyy käsittelemään dynaamisen taulukon avulla miten monta lämpötilaa tahansa.

Lämpötilojen tallentaminen dynaamiseen taulukkoon

1. Avaa Project-ikkuna ja valitse MyFixArray.frm. Tallenna MyFixArray-projektin jokainen tiedosto uudella nimellä, jotta alkuperäiset säilyvät.
2. Valitse File-valikosta komento Save MyFixArray.frm As. Kirjoita Save File As -valintaikkunaan **MyDynArray.frm** ja napsauta Save.
3. Valitse Project-ikkunasta MyFixArray.bas-moduuli ja File-valikosta komento Save MyFixArray.bas As. Kirjoita Save File As -valintaikkunaan **MyDynArray.bas** ja napsauta Save.
4. Valitse File-valikosta komento Save Project As, kirjoita **MyDynArray.vbp** ja napsauta Save.

Näytä lauseet -painike

5 Valitse Project-ikkunasta Module1 ja napsauta Näytä lauseet -painiketta, jotta näet moduulin sisällön Code-ikkunassa.

6 Poista taulukon esittelylauseesta numero 7, jotta Temperatures-taulukko muuttuu dynaamiseksi.

Lauseen pitää näyttää tällaiselta:

```
Public Temperatures() As Variant
```

7 Lisää vakiomoduuliin seuraava julkisen kokonaislukumuuttujan esittely:

```
Public Days As Integer
```

Tähän muuttujaan tallennetaan käyttäjän antama numero, joka määrää taulukon koon ohjelman suorituksen aikana.

Näytä objekti -painike

8 Sulje vakiomoduuli. Valitse Project-ikkunasta Form1, napsauta Näytä objekti -painiketta ja kaksoisnapsauta Syötä lämpötilat -painiketta. Muokkaa tapahtuma-aliohjelma cmdEnterTemps_Click seuraavan listauksen mukaiseksi. (Muutetut kohdat on lihavoitu.)

```
Cls
Days = InputBox("Kuinka monta päivää?", "Luo taulukko")
If Days > 0 Then ReDim Temperatures(Days)
Prompt$ = "Anna päivän ylin lämpötila."
For i% = 1 To Days
    Title$ = "Päivä  " & i%
    Temperatures(i%) = InputBox(Prompt$, Title$)
Next i%
```

Toinen ja kolmas lause kysyvät käyttäjältä, kuinka monen päivän lämpötilat hän haluaa tallentaa taulukkoon, sekä määrittävät dynaamisen taulukon koon tämän tiedon perusteella. If...Then-lause varmistaa, että päivien määrä on nollaa suurempi (taulukon koon määrittäminen nollaksi tai pienemmäksi aiheuttaa ajonaikaisen virheen). Days-muuttuja toimii myös For...Next-silmukan ylärajana.

9 Avaa objektiluettelo ja valitse cmdDisplayTemps-objekti. Muokkaa tapahtuma-aliohjelma cmdDisplayTemps_Click seuraavan listauksen mukaiseksi. (Muutetut kohdat on lihavoitu.)

```
Print "Ylimmät päivälämpötilat:"
Print
For i% = 1 To Days
    Print "Day "; i%, Temperatures(i%)
    Total! = Total! + Temperatures(i%)
Next i%
Print
Print "Ylimpien lämpötilojen keskiarvo:  "; Total! / Days
```

Days-muuttuja korvaa numeron 7 kahdessa kohdassa.

Tallenna projekti -painike

10 Sulje Code-ikkuna ja tallenna muutokset napsauttamalla Tallenna projekti -painiketta.

11 Käynnistä ohjelma napsauttamalla työkalurivin Käynnistä-painiketta.

12 Napsauta Syötä lämpötilat -painiketta ja valitse tallennettavien päivälämpötilojen määräksi 5.

13 Syötä viisi lämpötila-arvoa, kun ohjelma kysyy niitä.

14 Kun olet antanut kaikki lämpötilat, napsauta Näytä lämpötilat -painiketta.

Valmis DynArray-ohjelma on kansiossa \Vb5Sbs\Less10.

Ohjelma näyttää lomakkeessa viisi syöttämääsi lämpötilaa ja laskee niiden keskiarvon. Lomake näyttää suunnilleen tällaiselta:

15 Lopeta ohjelma napsauttamalla työkalurivin Lopeta-painiketta.

Lopeta-painike

Nyt olet oppinut tallentamaan rajoittamattoman määrän arvoja dynaamiseen taulukkoon ja käsittelemään niitä For...Next-silmukalla. Näistä taidoista on sinulle hyötyä, kun sinun pitää säilyttää suuria tietomääriä muistissa ohjelman suorituksen aikana. Seuraavassa harjoituksessa opit tallentamaan tällaisia tietoja tekstitiedostoihin ja tietokantoihin.

Lisätietoja: Moniulotteisten taulukoiden käyttö

Moniulotteinen taulukko sisältää useita taulukoita.

Visual Basicilla voidaan muodostaa yksiulotteisten taulukoiden eli listojen ohella *moniulotteisia taulukoita*. Moniulotteisten taulukoiden käsitteleminen voi olla aluksi hankalaa, ja monet niiden sovellukset ovat tämän kirjan aihealueen ulkopuolella. Kun olet oppinut käyttämään moniulotteisia taulukoita, säästät paljon aikaa ja vaivaa, jos ohjelmassa tarvitaan suuria tietomääriä.

Seuraavassa harjoituksessa tehtäväsi on luoda kaksiulotteinen taulukko, johon tallennetaan kuvitteellisen baseball-ottelun tulokset. Se muistuttaa aiemmin tässä harjoituksessa kuvattua 2-rivistä ja 9-sarakkeista taulukkoa. Kun olet oppinut hallitsemaan kaksiulotteiset taulukot, haluat ehkä kokeilla kolmi- ja neliulotteisia taulukoita saadaksesi vähän aivovoimistelua. (Tällaisia taulukoita käytetään usein tietokonegrafiikassa ja tieteellisissä sovelluksissa.)

Baseball-ottelun tulostaulun tallentaminen kaksiulotteiseen taulukkoon

Avaa projekti -painike

1. Napsauta työkalurivin Avaa projekti -painiketta.
2. Avaa Baseball.vbp-projekti kansiosta \Vb5Sbs\Less10.

   Ohjelma latautuu muistiin, ja projektin tiedostot näkyvät Project-ikkunassa.

Näytä objekti -painike

3. Napsauta Baseball.frm-lomaketta ja napsauta Näytä objekti -painiketta.

   Tulostaulu-lomake tulee näyttöön tällaisena:

Lomakkeessa on kaksi tekstiruutua ja kaksi komentopainiketta, joilla syötetään yhdeksästä pelivuorosta muodostuvan baseball-ottelun juoksujen määrät. Kotijoukkueena on Huitojat ja vierasjoukkueena Pelurit, mutta voit vaihtaa näiden tilalle suosikkijoukkueittesi nimet. Ohjelma näyttää tulokset taulussa käyttämällä Print-metodia ja tallentaa ne kaksiulotteiseen Scoreboard-nimiseen julkiseen taulukkoon.

4. Käynnistä ohjelma napsauttamalla työkalurivin Käynnistä-painiketta.

   Ohjelman suoritus alkaa, ja kohdistin vilkkuu Huitojat-tekstiruudussa, joka on lomakkeen vasemmassa alakulmassa.

Käynnistä-painike

Tekstiruudun yläpuolella näkyvä teksti ilmaisee, että ohjelma on valmis tallentamaan ensimmäisen vaihtovuoron tulokset.

5 Kirjoita **0** ja paina SARKAINNÄPPÄINTÄ.

Kohdistin siirtyy Pelurit-teksiruutuun.

6 Kirjoita **0** ja napsauta Seuraava sisävuoro -painiketta.

Lomakkeen tulostaulussa näkyy tasapelitulos 0–0.

7 Syötä muiden vaihtovuorojen tulokset, kunnes kaikki yhdeksän vuoroa ovat valmiit.

Voit napsauttaa Seuraava sisävuoro -painiketta useita kertoja, jos kahden peräkkäisen vuoron tulokset ovat samat.

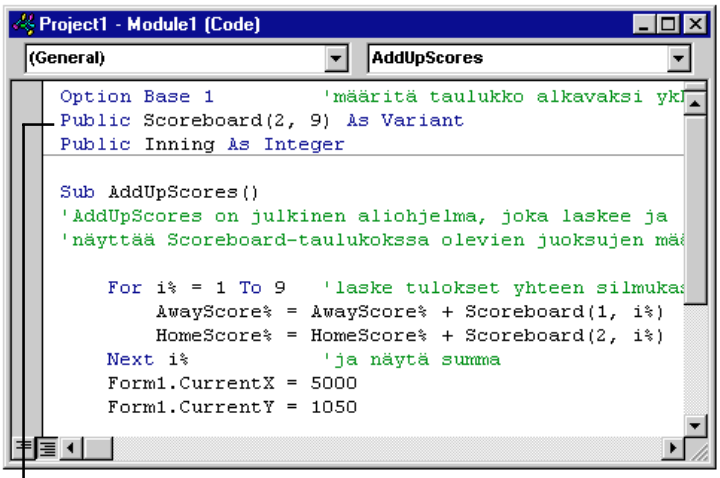
Kun olet valmis, lomakkeen pitäisi muistuttaa alla esitettyä kuvaa.

8 Lopeta ohjelma napsauttamalla Lopeta-painiketta.

9 Valitse Project-ikkunasta Baseball.bas-vakiomoduuli ja napsauta Näytä lauseet -painiketta.

Näytä lauseet –painike

Näyttöön tulee Code-ikkuna, joka sisältää vakiomoduulin esittelyosan ja AddUpScores-aliohjelman alkuosan. Nämä on esitetty seuraavan sivun kuvassa.



Scoreboard on julkinen kaksiulotteinen taulukko

Vakiomoduuli voi sisältää julkisia taulukoita, muuttujia ja aliohjelmia.

Vakiomoduulissa on neljä tärkeää osaa: Option Base -lause, julkisen kaksiulotteisen (2 x 9) Scoreboard-taulukon esittely, julkisen Inning-muuttujan esittely ja yleiskäyttöinen AddUpScores-aliohjelma. Scoreboard-taulukko on esitelty taulukoksi, jossa rivit edustavat kahta joukkuetta ja sarakkeet sisävuoroja. Taulukko muistuttaa rakenteeltaan yksinkertaista laskentataulukkoa. Inning-muuttujalla viitataan taulukon sarakearvoihin.

10 Tutki AdUpScores-aliohjelman lauseita vähän aikaa.

Aliohjelma laskee taulukkoon tallennettujen tulosten summat ja näyttää ne. Koska tämä julkinen aliohjelma on esitelty vakiomoduulissa, sitä voidaan kutsua kaikkialta ohjelmasta.

11 Sulje vakiomoduuli ja kaksoisnapsauta Baseball.frm-lomakkeen Seuraava sisävuoro -painiketta.

Tähän painikkeeseen liittyvä tapahtuma-aliohjelma siirtää ottelun tulokset joukkuekohtaisista tekstiruuduista Scoreboard-taulukkoon ja lopulta lomakkeen tulostauluun. Kussakin Scorebard-taulukon sijoituslauseessa tarvitaan kaksi indeksiä: toinen riviä ja toinen saraketta varten. Kohdistimen sijainti on huolellisesti laskettu siten, että tiedot tulevat oikeisiin paikkoihin tulostaulussa. CurrentX ja CurrentY ovat ominaisuuksia, jotka määrittävät kohdistimen koordinatit lomakkeessa. Niiden mittayksikkö on twip eli 1/1440 tuumaa (noin 0,0176 mm). Koska millimetrille mahtuu noin 57 twip-yksikköä, koordinaatisto on niin tiheä, että kohdistin voidaan sijoittaa erittäin tarkasti. Twip-koordinaatisto on erittäin tehokas, mutta vaatii kyllä hieman harjoittelua.

CurrentX- ja CurrentY-ominaisuudet sijoittavat kohdistimen tulostauluun.

12 Vieritä tapahtuma-aliohjelmaa Form_Load alaspäin Code-ikkunassa.

Show-metodi tarvitaan, jos Print-lause suoritetaan lomakkeen lataamisen aikana.

Form_Load alustaa Inning-muuttujan arvoksi 1 ja näyttää otsikon tulostaulussa. Aliohjelmassa tarvitaan Show-metodi ennen Print-metodia. Normaalisti Print-metodi pystyy näyttämään tiedot omin avuin, mutta koska lomakkeen lataaminen eli Form_load-tapahtuma on vielä kesken, tietojen näyttämiseen tarvitaan Show-metodia.

13 Katso, löydätkö Baseball-ohjelmasta muitakin ideoita omia ohjelmiasi varten, ja käynnistä se halutessasi uudelleen. Kun olet lopettanut, siirry seuraavaan harjoitukseen.

Jos haluat jatkaa seuraavaan harjoitukseen

➤ Jätä Visual Basic käyntiin ja siirry harjoitukseen 11.

Jos haluat lopettaa Visual Basicin

➤ Valitse File-valikosta komento Exit.

Jos näyttöön tulee Save-valintaikkuna, napsauta No.

Harjoituksen tiivistelmä

Tehtävä	Ohje

Käsitellä kokoelman objekteja	Kirjoita For Each…Next -silmukka, joka käy läpi kokoelman jäsenet. Esimerkki:
	`For Each Ctrl In Controls`
	`  Ctrl.Visible = False`
	`Next Ctrl`

Siirtää Controls-kokoelman objekteja vasemmalta oikealle lomakkeen poikki	Muokkaa kokoelman kunkin objektin Control.Left-ominaisuutta For Each…Next -silmukassa. Esimerkki
	`For Each Ctrl In Controls`
	`  Ctrl.Left = Ctrl.Left + 200`
	`Next Ctrl`

Käsitellä kokoelman objekti erityisellä tavalla	Anna objektin Tag-ominaisuudelle helposti tunnistettava arvo ja testaa se For Each…Next -silmukassa. Esimerkki:
	`For Each Ctrl In Controls`
	`  If Ctrl.Tag <> "Button" Then`
	`  Ctrl.Left = Ctrl.Left + 200`
	`  End If`
	`Next Ctrl`

Tehtävä	Ohje
Luoda julkinen taulukko	Esittele taulukko vakiomoduulissa käyttäen Public-avainsanaa. Esimerkki: `Public Employees(9) As String`
Luoda paikallinen taulukko	Esittele taulukko tapahtuma-aliohjelmassa käyttäen Static-avainsanaa. Esimerkki: `Static Employees(9) As String`
Sijoittaa taulukkoon arvo	Määritä taulukon nimi, alkion indeksi ja arvo. Esimerkki: `Employees(5) = "Leslie"`
Indeksoida kaikki taulukot alkaen ykkösestä	Kirjoita vakiomoduuliin Option Base -lause. Esimerkki: `Option Base 1`
Poistaa Print-lauseiden tulokset lomakkeesta	Käytä Cls-metodia.
Luoda dynaaminen taulukko	Määritä taulukon nimi ja tyyppi ohjelmointivaiheessa, mutta jätä alkioiden määrä avoimeksi. Kun ohjelma on käynnissä, määritä taulukon koko ReDim-lauseella. Esimerkki: `ReDim Temperatures(Days)`

Käsitellä taulukon alkioita	Kirjoita For...Next-silmukka, joka valitsee taulukosta alkioita laskurimuuttujan avulla. Esimerkki:
	`For i% = 1 To 7`
	`    Total! = Total! + Temperatures(i%)`
	`Next i%`

Sijoittaa kohdistin lomakkeeseen (Print- ja muita metodeja varten)	Määritä lomakkeen CurrentX- ja CurrentY-ominaisuudet. Nämä edustavat kohdistimen *x*,*y*-koordinaatteja twip-yksiköinä.

Kun tarvitset tietoja aiheesta	Valitse Visual Basicin Help-valikosta Microsoft Visual Basic Help Topics, valitse Index-välilehti ja...
Controls-kokoelma	Kirjoita **collections, controls**.
Printers-kokoelma	Kirjoita **collections, printers**.
For Each…Next -silmukat	Kirjoita **For**.
Taulukoiden luominen ja käyttö	Kirjoita **arrays**.
Kohdistimen sijainnin määrittäminen lomakkeessa	Kirjoita **CurrentX property**.

Tietoja seuraavasta harjoituksesta

Seuraavassa harjoituksessa opit luomaan tekstitiedostoja ja näyttämään niiden sisältöjä ohjelmallisesti sekä järjestämään kiintolevysi sisällön tiedostojärjestelmän komennoilla. Opit myös avaamaan ja muokkaamaan tietokantoja Visual Basicin Data-objektilla.

Harjoitus

11

Tekstitiedostojen ja tietokantojen käsittely

Arvioitu aika
50 min.

Tässä harjoituksessa opit

- näyttämään tekstitiedoston sisällön tekstiruutuobjektilla
- tallentamaan muistiinpanoja tekstitiedostoon
- muodostamaan Data-objektista tietokannan katseluohjelman
- etsimään tietokannasta tietoja
- lisäämään ja poistamaan tietokannan tietueita
- tekemään tiedostoista varmistuskopioita FileCopy-lauseella.

Tässä harjoituksessa näytän, miten voit käsitellä tekstitiedostoihin ja tietokantoihin tallennettuja tietoja. Opit avaamaan tekstitiedoston ja näyttämään sen sisällön tekstiruutuobjektissa sekä luomaan tekstitiedoston levylle. Opit myös avaamaan tietokantoja dataobjektilla, etsimään tietoja sekä lisäämään ja poistamaan tietueita. Visual Basic on erityisesti suunniteltu tietokantojen käyttöliittymien muodostamista varten. Jos haluat mukauttaa tai parannella tietokantoja, joita olet tehnyt toisella sovelluksella, esimerkiksi Microsoft Accessilla, pääset heti alkuun.

Tekstitiedoston sisällön näyttäminen tekstiruutuobjektilla

Yksinkertaisin tapa näyttää tekstitiedoston sisältö ohjelmassa on käyttää tekstiruutuobjektia. Voit tehdä erikokoisia tekstiruutuja, ja jos tekstitiedoston sisältö ei mahdu ruutuun, lisää vierityspalkit, joilla käyttäjä voi selata tekstiä. Kun ohjelman pitää lukea tekstitiedosto tekstiruutuun, siinä tarvitaan kolme lausetta ja yksi funktio. Nämä avainsanat on lueteltu seuraavassa talukossa, ja niitä käytetään tämän harjoituksen ensimmäisessä ohjelmassa.

Visual Basicissa on neljä avainsanaa tekstitiedostoja varten.

Avainsana	Kuvaus
Open	Avaa tekstitiedoston syöttämistä tai tulostamista varten.
Line Input	Lukee rivin tekstitiedostosta.
EOF	Tarkistaa, onko tekstitiedosto lopussa.
Close	Sulkee tekstitiedoston.

Tekstitiedoston avaaminen syöttöä varten

Tekstitiedostossa on selväkielisiä merkkejä.

Tekstitiedostossa on yksi tai useita rivejä lukuja, sanoja tai merkkejä. Se eroaa asiakirjatiedostosta, jossa on myös muotoilukoodeja, ja suoritettavasta tiedostosta, jossa on käskyjä käyttöjärjestelmälle. Microsoft Windows Resurssienhallinnassa (Explorer) käytetään tekstitiedostoista myös nimeä "tekstiasiakirja", ja niiden tarkenne on .txt, .ini, .log, .inf, .dat tai .bat. Koska tekstitiedosto sisältää vain tavallisia, selväkielisiä merkkejä, voit näyttää ne helposti tekstiruutuobjektissa.

Yleinen valintaikkunaobjekti näyttää yleisen Avaa-valintaikkunan.

Voit antaa käyttäjän valita avattavan tiedoston käyttämällä yleistä valintaikkunaobjektia, joka kysyy tiedoston polkunimen. Yleiset valintaikkunaobjektit tuntevat ShowOpen-metodin, joka näyttää Avaa-valintaikkunan (Open). Ohjelma saa käyttäjän valitseman tiedoston polkunimen FileName-ominaisuuden kautta, ja se voi avata tiedoston tämän nimen perusteella. Yleinen valintaikkunaobjekti ei avaa tiedostoa, vaan ainoastaan lukee polkunimen.

Open-lause

Kun olet saanut selville tiedoston polkunimen, avaa tiedosto Open-lauseella. Sen muoto on tällainen:

```
Open polkunimi For tila As #tiedostonumero
```

Seuraavat argumentit ovat tärkeitä:

- *Polkunimi* on sallittu Microsoft Windows -polkunimi.
- *Tila* on avainsana, joka ilmaisee, miten tiedostoa käytetään. Tässä harjoituksessa käytetään syöttötilaa (Input) ja tulostustilaa (Output).
- *Tiedostonumero* on kokonaisluku 1–255.

Tiedostonumero liitetään tiedostoon avaamisen yhteydessä. Käytä sitä ohjelmassa aina, kun viittaat avattuun tiedostoon. Tiedostonumeroissa ei ole mitään ihmeellistä; Visual Basic pitää niiden perusteella kirjaa ohjelmalla avatuista tiedostoista.

Tavanomainen yleistä valintaikkunaobjektia käyttävä Open-lause näyttää tällaiselta:

```
Open CommonDialog1.FileName For Input As #1
```

Tässä CommonDialog1.FileName-ominaisuus edustaa tiedoston nimeä, Input on tila ja 1 on tiedostonumero.

HUOMAA Tällaisella käskyllä avattavia tekstitiedostoja sanotaan *peräkkäistiedostoiksi*, koska niiden sisältö on luettava alusta loppuun. Sen sijaan tietokantatiedostoa voidaan lukea missä järjestyksessä tahansa.

Seuraavaksi voit kokeilla, miten tekstitiedosto avataan käyttäen yleistä valintaikkunaobjektia ja Open-lausetta. Samalla näet, miten tekstitiedoston sisältö näytetään tekstiruudussa Line Input- ja EOF-avainsanoilla ja miten tiedosto suljetaan Close-avainsanalla.

Text Browser-ohjelman suorittaminen

1. Käynnistä Visual Basic, ellei se ole vielä käynnissä.
2. Napsauta työkalurivin Avaa projekti -painiketta.

Avaa projekti -painike

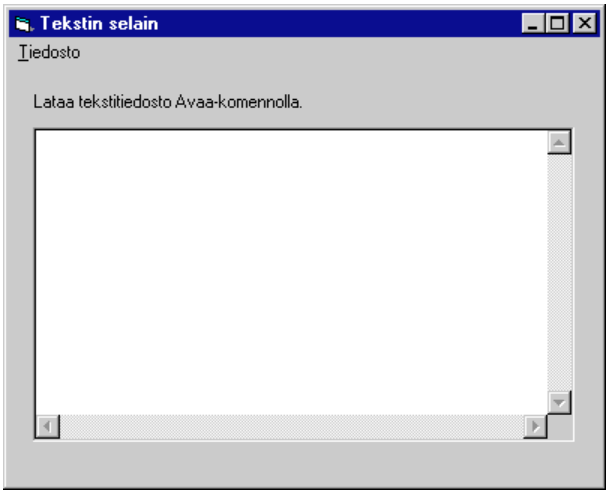
3. Valitse kansio \Vb5Sbs\Less11 ja kaksoisnapsauta tiedostoa ShowText.

   ShowText-ohjelma latautuu ohjelmointiympäristöön.

Valmis Show-Text-ohjelma on kansiossa \VB5Sbs\Less11.

4. Ellei ShowText-lomake ole vielä näkyvissä, valitse se Project-ikkunasta ja Napsauta Näytä objekti -painiketta.

   ShowText-lomake tulee esiin seuraavan sivun kuvan mukaisena.



Lomakkeessa on suuri tekstiruutuobjekti vierityspalkkeineen. Siinä on myös yleinen valintaikkunaobjekti, ohjelman käyttöä kuvaava tekstiobjekti sekä File-valikko, jossa on komennot Open, Close ja Exit. Objektien ominaisuudet on määritetty seuraavan taulukon mukaisesti (huomaa varsinkin tekstiruudun asetukset).

Objekti	Ominaisuus	Asetus
txtFile	Enabled	False
	Multiline	True
	Name	txtFile
	ScrollBars	3 - Both
	Text	(Tyhjä)
mnuItemClose	Enabled	False
	Name	mnuItemClose
lblFile	Caption	”Lataa tekstitiedosto Avaa-komennolla.”
	Name	lblFile
Form1	Caption	”Tekstin selain”

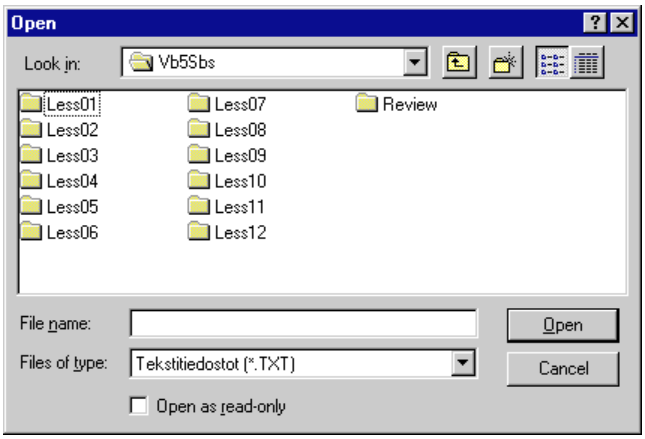
Käynnistä-painike

5 Napsauta työkalurivin Käynnistä-painiketta.

Tekstin selain -ohjelma käynnistyy.

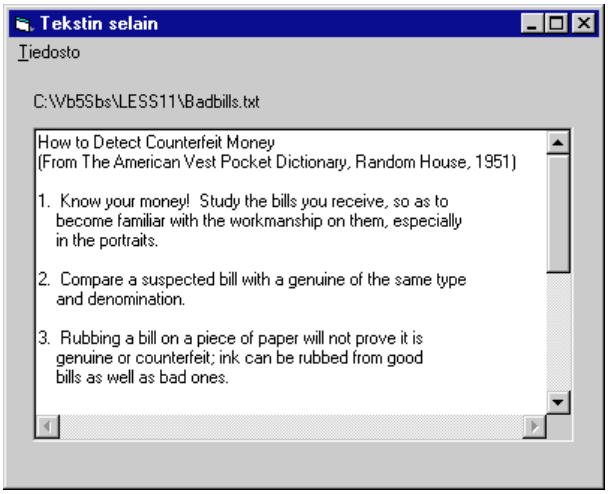
6 Valitse ohjelman File-valikosta komento Open.

Open-valintaikkuna avautuu seuraavan kuvan näköisenä.



7 Valitse Open-valintaikkunassa kansio \Vb5Sbs\Less11 ja kaksoisnapsauta tiedostoa Badbills.txt.

Tekstiruutuun ilmestyy tekstitiedosto Badbills, jossa on väärennettyä rahaa koskeva artikkeli vuodelta 1951.



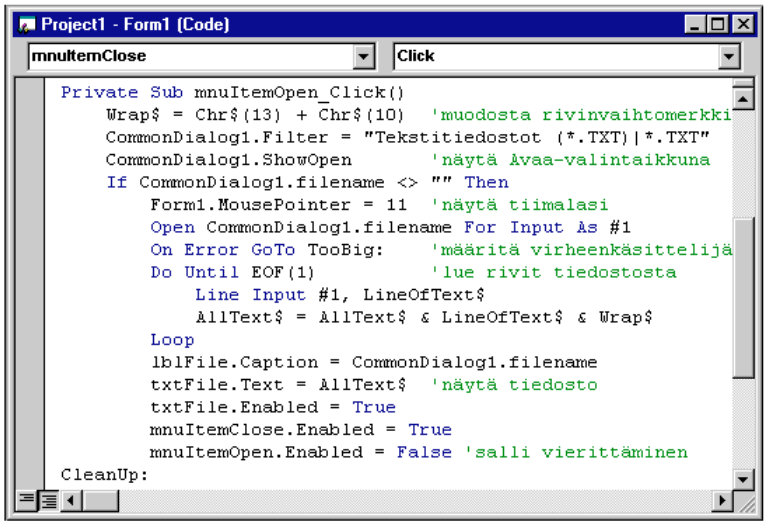
8 Napsauta vierityspalkkeja, jotta voit lukea koko tiedoston.

9 Kun olet lukenut tekstin, sulje tiedosto valitsemalla Tiedosto-valikosta komento Sulje ja lopeta ohjelma valitsemalla komento Lopeta.

Ohjelman suoritus päättyy, ja ohjelmointiympäristö palaa näkyviin. Tarkastele seuraavaksi ohjelman kahta tärkeää tapahtuma-aliohjelmaa.

ShowText-ohjelman tarkasteleminen

1. Valitse Tekstin selain -lomakkeen Tiedosto-valikosta komento Avaa.
   Tapahtuma-aliohjelma mnuItemOpen_Click ilmestyy Code-ikkunaan.
2. Suurenna Code-ikkunaa, jotta näet suuremman osan aliohjelmasta.
   Ikkunan pitäisi näyttää suunnilleen tällaiselta:



Tämä aliohjelma suorittaa alla luetellut tehtävät. (Vastaavat Visual Basic -lauseet on ympäröity sulkein.)

- Se kysyy tiedoston polkunimen käyttäen yleistä valintaikkunaobjektia.
- Se avaa tiedoston lukemista varten (Open...For Input).
- Se kopioi tiedoston rivi kerrallaan merkkijonoon AllText$ (Line Input).
- Se kopioi rivejä, kunnes tiedosto loppuu (EOF) tai merkkijono tulee täyteen. AllText$-merkkijonossa on tilaa 64 kilotavulle tekstiä.
- Se näyttää AllText$-merkkijonon tekstiruudussa ja lisää vierityspalkit.
- Se käsittelee mahdolliset virhetilanteet (On Error GoTo).
- Se päivittää Tiedosto-valikon komennot ja hiiren osoittimen ja sulkee tiedoston (Close).

3. Mieti hieman, miten tapahtuma-aliohjelman lauseet toimivat — varsinkin avainsanat Open, Line Input, EOF ja Close.

Jos haluat lisätietoja näistä lauseista, katso Visual Basicin ohjeita. Aliohjelman virheenkäsittelijä TooBig näyttää ilmoituksen ja keskeyttää lataamisen, jos käyttäjä on valinnut tiedoston, joka on suurempi kuin 64 kilotavua. Tämä on tarpeen siksi, että tekstiruutuobjektin merkkijonon kokorajoitus on 64 kilotavua.

Jos valitset monta sivua pitkän tiedoston, sen lataaminen kestää jonkin aikaa. Siksi osoitin muutetaan tiimalasiksi MousePointer-ominaisuuden avulla siihen asti, kunnes tiedoston sisältö tulee näyttöön. Käyttäjälle kannattaa aina ilmaista jollakin tavalla, että hänen pitää odottaa tehtävän suorittamista muutama sekunti.

4. Avaa objektiluettelo ja valitse mnuItemClose, jotta näet tapahtuma-aliohjelman mnuItemClose_Click.

   Tämä aliohjelma suoritetaan, kun Sulje-komento valitaan. Se tyhjentää tekstiruudun, poistaa Sulje-komennon käytöstä, ottaa Avaa-komennon käyttöön ja poistaa tekstiruudun käytöstä.

5. Kun olet tutkinut ShowText-ohjelman lauseita tarpeeksi, sulje Code-ikkuna.

Nyt voit käyttää tätä pelkistettyä ohjelmaa mallina monimutkaisemmille apuohjelmille, jotka käsittelevät tekstitiedostoja. Seuraavaksi opit kirjoittamaan tekstiä tekstiruutuun ja tallentamaan sen levylle.

Tekstitiedoston luominen levylle

Käytä Open-lauseessa avainsanoja For Output, kun luot tiedoston.

Luodessasi tekstitiedoston Visual Basicilla tarvitset useita edellisestä esimerkistä tuttuja objekteja ja avainsanoja. Tiedostojen luomisesta ja tietojen tallentamisesta niihin on sinulle apua, jos aiot laatia raportteja tai lokeja, tallentaa tärkeitä arvoja tai laskutoimitusten tuloksia tai tehdä omaan käyttöösi erikoisen tekstinkäsittelyohjelman tai tekstieditorin. Tässä on esitetty päävaiheet, jotka ohjelman pitää suorittaa:

1. Kysy käyttäjältä tietoja tai suorita laskutoimituksia tai molempia.
2. Sijoita tulokset yhteen tai useaan muuttujaan. Voisit esimerkiksi sijoittaa tekstiruudun sisällön muuttujaan nimeltä InputForFile$.
3. Kysy käyttäjältä polkunimi yleisessä Tallenna nimellä -valintaikkunassa (Save As). Voit avata sen ShowSave-metodilla.
4. Käytä valintaikkunasta saamaasi polkunimeä tiedoston avaamiseen tulostusta varten (Open...For Output).

Print # -lause lähettää tulokset määritettyyn tiedostoon.

5. Tallenna tiedot avoimeen tiedostoon Print # -lauseilla (Print #).
6. Sulje tiedosto, kun olet valmis (Close).

Seuraavassa ohjelmassa esitetään, miten tekstiruudusta, yleisistä valintaikkuna-objekteista sekä Open-, Print #- ja Close-lauseista kootaan yksinkertainen ohjelma muistiinpanojen tekemistä varten. Voit kirjoittaa ohjelmalla tekstiä ja lisätä siihen aikaleiman.

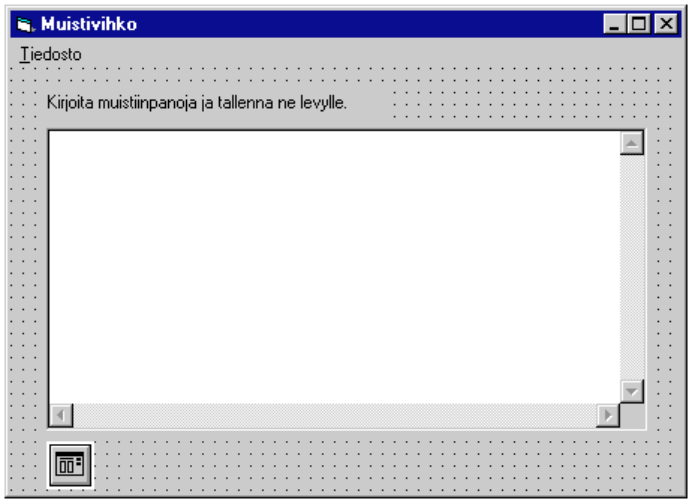
QNote-ohjelman suorittaminen

Avaa projekti -painike

1. Napsauta työkalurivin Avaa projekti -painiketta.
2. Valitse kansio \Vb5Sbs\Less11 ja kaksoisnapsauta projektia QNote.

   QNote-ohjelma latautuu ohjelmointiympäristöön.
3. Ellei Muistivihko-lomake ole näkyvissä, valitse se Project-ikkunasta ja napsauta Näytä objekti -painiketta.

   Muistivihko-lomake tulee esiin seuraavan kuvan näköisenä:



Lomakkeessa on suuri tekstiruutuobjekti, jonka reunoissa on vierityspalkit. Se sisältää myös yleisen valintaikkunaobjektin, ohjelman käyttöohjeen tekstiobjektina sekä Tiedosto-valikon, jossa on komennot Tallenna nimellä, Lisää päivämäärä ja Lopeta. Olen määrittänyt objektien ominaisuudet seuraavan luettelon mukaisesti.

Objekti	Ominaisuus	Asetus
txtNote	Multiline	True
	Name	txtNote
	ScrollBars	3 - Both
	Text	(Tyhjä)
Label1	Caption	”Kirjoita muistiinpanoja ja tallenna ne levylle.”
Form1	Caption	”Muistivihko”

Käynnistä-painike

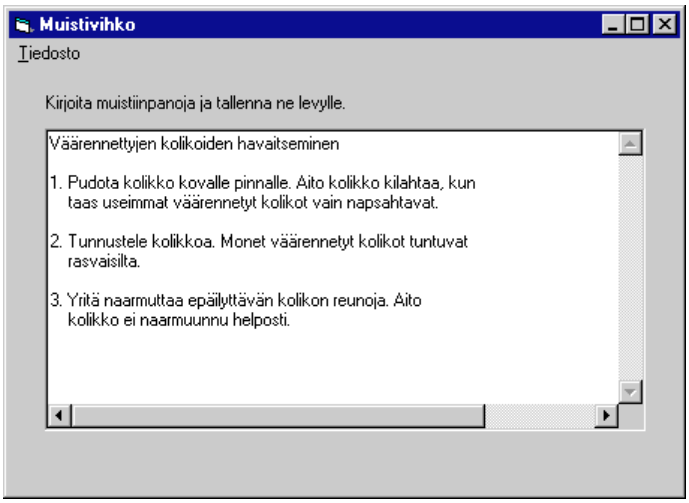
4 Napsauta työkalurivin Käynnistä-painiketta.

5 Kirjoita tekstiruutuun seuraava teksti tai jotakin muuta:

Väärennettyjen kolikoiden havaitseminen

1. **Pudota kolikko kovalle pinnalle. Aito kolikko kilahtaa, kun taas useimmat väärennetyt kolikot vain napsahtavat.**
2. **Tunnustele kolikkoa. Monet väärennetyt kolikot tuntuvat rasvaisilta.**
3. **Yritä naarmuttaa epäilyttävän kolikon reunoja. Aito kolikko ei naarmuunnu helposti.**

Valmiin ikkunan pitäisi näyttää tällaiselta:

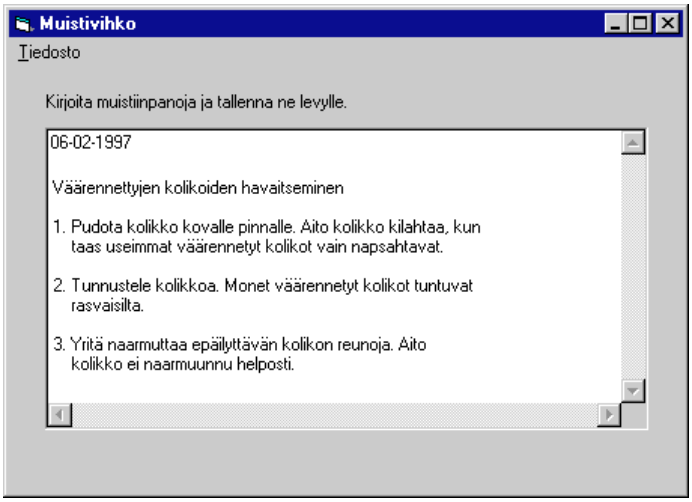


HUOMAA Voit kopioida tekstiä Windowsin Leikepöydältä tekstiruutuun painamalla CTRL+V. Jos haluat kopioida tekstiä tekstiruudusta Leikepöydälle, valitse se hiirellä ja paina CTRL+C.

Kokeile Tiedosto-valikon komentoja.

6 Valitse Tiedosto-valikosta komento Lisää päivämäärä.

Nykyinen päivämäärä ilmestyy tekstiruudun ensimmäiselle riville:



Lisää päivämäärä -komento on kätevä tapa lisätä tiedostoon päivämäärä. Siitä on hyötyä, jos pidät päiväkirjaa tai lokia.

7 Valitse Tiedosto-valikosta komento Tallenna nimellä.

8 Valitse Tallenna nimellä -valintaikkunasta (Save As) kansio \Vb5Sbs\Less11, ellei sitä ole vielä valittu. Kirjoita tiedoston nimeksi **Badcoins.txt** ja napsauta Save.

Kirjoittamasi teksti tallentuu uuteen tiedostoon nimeltä Badcoins.txt. Nyt voit lopettaa ohjelman.

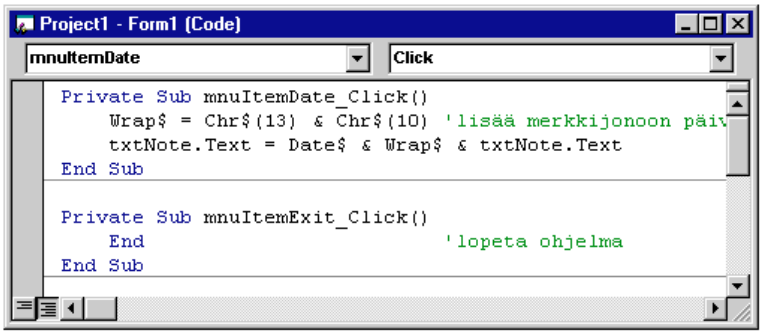
9 Valitse Tiedosto-valikosta komento Lopeta.

Ohjelman suoritus päättyy, ja ohjelmointiympäristö palaa näkyviin. Tutkitaan ohjelman tapahtuma-aliohjelmia hieman.

QNote-ohjelman tarkasteleminen

1 Napsauta Muistivihko-lomakkeen Tiedosto-valikon komentoa Lisää päivämäärä.

Tapahtuma-aliohjelma mnuItemDate_Click ilmestyy Code-ikkunaan seuraavan kuvan kaltaisena:



Date$-palauttaa nykyisen päivämäärän.

Tämä aliohjelma lisää tekstiruutuun nykyisen päivämäärän liittämällä peräkkäin Date$-funktion palauttaman päivämäärän, rivinvaihdon ja Text-ominaisuuden. Voisit lisätä tekstiruudun sisältöön nykyisen kellonajan tai muita tietoja samalla menetelmällä.

2 Selvitä itsellesi, miten merkkijonojen liittäminen toimii, ja avaa sitten tapahtuma-aliohjelma mnuItemSave_Click.

Print # -lause saa ensimmäiseksi argumentikseen tiedostonumeron.

Aliohjelma avaa Tallenna nimellä -valintaikkunan käyttäen yleistä valintaikkunaobjektia, avaa tiedoston tulostusta varten numerolla 1, kirjoittaa txtNote.Text-ominaisuuden arvon levylle Print # -lauseella ja sulkee tekstitiedoston. Huomaa erityisesti lause

```
Print #1, txtNote.Text
```

joka sijoittaa tekstiruudun koko sisällön avoimeen tiedostoon. Print # -lause on samankaltainen kuin Print-metodi, paitsi että tulostustiedot ohjautuvat määritettyyn tiedostoon näytön tai tulostimen sijasta. Tässä on tärkeää huomata, että tiedoston koko sisältö on merkkijonossa txtNote.Text.

3 Tarkastele Open-, Print #- ja Close-lauseita ja sulje Code-ikkuna.

QNote-ohjelma on nyt loppuun käsitelty. Näissä kahdessa esimerkissä opit niin paljon, että osaat auttavasti käsitellä tekstitiedostoja. Löydät lisätietoja peräkkäistiedostoista katsomalla Visual Basicin ohjeita.

Tietokantojen hallinta

Kuten harjoituksessa 2 näit, tietokanta on järjestelmällinen tietokokoelma, joka on tallennettu elektronisesti tiedostoon. Voit luoda tehokkaita tietokantoja useilla tietokantaohjelmilla, joita ovat esimerkiksi Microsoft Access, Microsoft FoxPro, Btrieve, Paradox ja dBase. Voit myös avata Open Database Connectivity (ODBC) -määrityksen mukaisia asiakas-palvelintietokantoja, esimerkiksi Microsoft SQL Server -tietokantoja.

Visual Basic osaa lukea ja kirjoittaa eri-muotoisia tieto-kantoja.

Jos käsittelet tietokantoja usein, varsinkin yllä kuvattuja tyyppejä, Visual Basic on harkinnan arvoinen väline tietojen muotoilemiseen ja näyttämiseen. Koska Visual Basic perustuu samaan tietokantatekniikkaan kuin Microsoft Access (tietokantaytimeen nimeltä Microsoft Jet), saat aikaan omia tietokantaohjelmia kirjoittamalla vain muutaman ohjelmarivin.

Tässä jaksossa opit, miten Visual Basicin dataobjektilla käsitellään tietokantaa Biblio.mdb, joka sisältyy Visual Basiciin ja Accessiin. Biblio.mdb on Access-tietokanta, joka sisältää tietokantaohjelmointia käsittelevien kirjojen nimiä. Kerron, miten voit näyttää tietokannan useita tietueita, etsiä tietoja, lisätä uusia tietueita, poistaa turhia tietueita ja kopioida tiedostoja. Kun olet käynyt läpi seuraavat tehtävät, osaat soveltaa taitojasi omiin tietokantaprojekteihin.

Mukautettujen tietokantasovellusten luominen

Mukautetut tietokantasovel-lukset näyttävät tietokannan tietueet ja ken-tät sopivassa muodossa.

Mukautettu tietokantasovellus on ohjelma, joka lukee kenttiä ja tietueita tieto-kannasta ja esittää ne tavalla, joka sopii tietylle käyttäjäryhmälle. Esimerkiksi yleisessä kirjastossa voitaisiin tehdä kirjatietokannasta erikoisversio tutkijoita varten. Sovellus sisältää yleensä erilaisia komentoja, joilla käyttäjä voi muodostaa katselusuotimia, etsiä, tulostaa, lisätä ja poistaa tietueita sekä tehdä tietokannasta varmistuskopioita. Eräät tietokannat on suunniteltu ja niitä on kehitelty siten, että niiden käyttäminen alkuperäisessä muodossa tai ympäristössä on hankalaa. Visual Basicilla voit tehdä oman tietokantasovelluksen, joka näyttää ainoastaan käyttäjien tarvitsemat tiedot ja komennot.

Tietokannan sisällön näyttäminen sidotuilla objekteilla

Sidotut objektit käsittelevät tietokannan sisältöä auto-maattisesti.

Useimmat Visual Basicin työkalurivin objektit pystyvät sellaisinaan näyttämään tietokannan tietoja. Tällaisia objekteja sanotaan *sidotuiksi objekteiksi*. Objekti on sidottu tietokantaan, kun sen DataSource-ominaisuus sisältää kelvollisen tietokannan nimen ja kun sen DataField-ominaisuus viittaa kelvolliseen tauluun tietokannassa. *Taulu* on kenttien ja tietueiden ryhmä, jonka sinä tai joku muu on määrittänyt tietokantaa luotaessa. Kuten opit harjoituksessa 2, voit kytkeä Visual Basic -ohjelman tietokantaan dataobjektilla. Yhteyden muodostamisen jälkeen voit näyttää tietokannan sisällön käyttäen objekteja, jotka on tehty seuraavan taulukon mukaisilla työkaluilla.

Objekti	Kuvaus
	Valintaruutu
	Avattava luetteloruutu
	Kuva
	Teksti
	Luetteloruutu
	Kuvakehys
	Tekstiruutu

Tietojen näyttäminen tekstiruutuobjekteilla

Seuraava ohjelma käyttää yhtä dataobjektia ja neljää tekstiruutuobjektia näyttäessään neljä Biblio.mdb-tietokannan kenttää. Siitä näet, miten voit laatia oman tietokantasovelluksen, joka näyttää vain haluamasi tiedot. Dataobjektin ReadOnly-ominaisuudeksi on määritetty True, jotta tietokannan tietoja voidaan katsella mutta ei muuttaa. Jos haluat antaa käyttäjille mahdollisuuden tehdä muutoksia tietokantaan, määritä Properties-ikkunassa ReadOnly-ominaisuudeksi False.

HUOMAA Jotta Biblio.mdb-tietokanta säilyisi muuttumattomana, tee siitä kopio Windows Resurssienhallinnalla (Explorer) ennen kuin aloitat seuraavan tehtävän.

DataView-ohjelman suorittaminen

Avaa projekti -painike

Näytä objekti -painike

1. Napsauta työkalurivin Avaa projekti -painiketta.
2. Valitse kansio \Vb5Sbs\Less11 ja kaksoisnapsauta projektia DataView.vbp.

   DataView-ohjelma latautuu ohjelmointiympäristöön.
3. Ellei Tietokannan selaus -lomaketta näy, valitse se Project-ikkunasta ja napsauta Näytä objekti-painiketta.

   Tietokannan selaus -lomake tulee näkyviin tällaisena:

Lomakkeessa on tietoja ohjelmasta, piirros, dataobjekti, useita tekstejä ja tekstiruutuja ja komentopainike. Seuraavan taulukon mukaiset ominaisuudet on määritetty tiedonsiirtoon liittyville objekteille eli data- ja tekstiruutuobjektille. Löydät muiden objektien ominaisuudet Properties-ikkunasta.

Objekti	Ominaisuus	Asetus
datBiblio	Caption	”Biblio.mdb”
	Connect	Access
	DatabaseName	”c:\program files\dev-studio\vb\biblio.mdb”
	Name	datBiblio
	ReadOnly	True
	RecordsetType	0 - Table
	RecordSource	Titles
txtTitle	DataField	Title
	DataSource	datBiblio
	Name	txtTitle
	Text	(Tyhjä)
txtInfo	DataField	Description
	DataSource	datBiblio
	Name	txtInfo
	Text	(Tyhjä)

Objekti	Ominaisuus	Asetus
txtISBN	DataField	ISBN
	DataSource	datBiblio
	Name	txtISBN
	Text	(Tyhjä)
txtYear	DataField	Year Published
	DataSource	datBiblio
	Name	txtYear
	Text	(Tyhjä)

Dataobjektin RecordSource-ominaisuudeksi on määritetty Titles ja DatabaseName-ominaisuudeksi "c:\program files\devstudio\vb\biblio.mdb . Neljän tekstiruudun DataSource-ominaisuudet ovat samat (datBiblio) ja DataField-ominaisuudet viittaavat tietokannan eri kenttiin. Näin muodostuu yhteys levyllä olevan tietokannan, ohjelman dataobjektin ja lomakkeen tekstikenttien välille. Muut ominaisuudet vain hienosäätävät näitä perusasetuksia.

HUOMAA Jos oma Biblio.mdb-tietokantasi ei ole kansiossa C:\Program Files\DevStudio\Vb (oletuspaikka), kirjoita dataobjektin DatabaseName-ominaisuuteen oikea polkunimi.

Käynnistä-painike

4 Napsauta työkalurivin Käynnistä-painiketta.

DataView-ohjelma käynnistyy ja näyttää Biblio.mdb-tietokannan Titles-taulun ensimmäisen tietueen lomakkeessa.

5. Napsauta dataobjektin sisempää oikealle osoittavaa nuolta, jotta näet Titles-taulun toisen tietueen.

   Lomakkeeseen ilmestyy kirjan *The dBASE Programming Language* tietue. Kaikki neljä tekstikenttää päivittyvät joka kerta siirtyessäsi tietueesta toiseen.

6. Napsauta ulompaa oikealle osoittavaa nuolta, jotta näet Titles-taulun viimeisen tietueen.

   Lomakkeeseen ilmestyy kirjan *1988 National Database and 4-5 Generation Language* tietue.

7. Lopeta ohjelma napsauttamalla Lopeta-painiketta.

   DataView-ohjelman suoritus päättyy.

Tässä ohjelmassa on vain yksi lause (End), mutta se esittää runsaasti tietoja. Ohjelman etu on se, että se näyttää tietokannasta vain tarvittavat kentät. Voit koota dataobjektista ja useasta tietoihin sidotusta tekstiruudusta tehokkaan tietokantasovelluksen.

Recordset-objektin käyttö

Recordset-objekti edustaa tietoja, joita ohjelmalla käsitellään.

DataView-ohjelmassa käytettiin dataominaisuutta nimeltä RecordsetType, joka muodosti tietokannan tiedoista taulun. Visual Basicissa *Recordset* on objekti, joka edustaa ohjelmalla käsiteltävää tietokannan osaa. Se sisältää erikoisominaisuuksia ja -metodeja, joilla etsitään, lajitellaan, lisätään ja poistetaan tietueita. Seuraava tehtäväsi on etsiä Biblio-mdb-tietokannasta kirjojen nimiä Recordset-objektilla.

Tietojen etsiminen Biblio.mdb-tietokannasta

Ennen kuin muokkaat ohjelmaa, tallenna alkuperäinen uudella nimellä.

1. Valitse File-valikosta komento Save DataView.frm As. Tallenna DataView-lomake kansioon \Vb5Sbs\Less11 nimellä **MyFindData.frm**.
2. Valitse File-valikosta komento Save Project As. Tallenna projekti nimellä **MyFindData.vbp**.
3. Napsauta Komentopainike-työkalua ja lisää lomakkeen vasempaan alakulmaan komentopainikeobjekti.

Komentopainike-työkalu

4. Määritä komentopainikeobjektin ominaisuudet näin:

Objekti	Ominaisuus	Asetus
Command1	Caption	"Etsi"
	Name	cmdFind

5. Kaksoisnapsauta Etsi-komentopainiketta, jotta tapahtuma-aliohjelma cmdFind_Click ilmestyy Code-ikkunaan.

6 Kirjoita tapahtuma-aliohjelmaan seuraavat ohjelmalauseet:

Seek-metodi etsii sopivan tietueen.

```
prompt$ = "Kirjoita kirjan täydellinen nimi."
'lue teksti, jota etsitään Title-kentästä
SearchStr$ = InputBox(prompt$, "Kirjan haku")
datBiblio.Recordset.Index = "Title"      'käytä nimeä
datBiblio.Recordset.Seek "=", SearchStr$ 'ja etsi
If datBiblio.Recordset.NoMatch Then      'ellei sopivaa löydy,
    datBiblio.Recordset.MoveFirst        'siirry 1. tietueeseen
End If
```

Aliohjelma avaa valintaikkunan, jossa se kysyy käyttäjältä etsittävän merkkijonon (SearchStr$), ja etsii merkkijonoa Seek-metodilla Title-kentästä alkaen tietokannan alusta. Se pysähtyy, kun merkkijono löytyy tai tietokanta loppuu. Ellei sopivaa kenttää löydy, ohjelma antaa ilmoituksen ja näyttää Recordset-objektin ensimmäisen tietueen. Aliohjelmassa on käytetty seuraavia Recordset-ominaisuuksia ja -metodeja:

Recordset-ominaisuus tai -metodi	Kuvaus
Index	Ominaisuus, jolla valitaan tietokannan tutkittavan kenttä.
Seek	Metodi, jolla etsitään tietue. Merkkijonoa voidaan verrata tietokannassa olevaan tekstiin yhtäläisyysoperaattorin = lisäksi operaattoreilla >=, >, <= ja <.
NoMatch	Ominaisuus, jonka arvo on True, jos etsittyä tietoa ei löytynyt.
MoveFirst	Metodi, joka valitsee Recordset-objektin ensimmäisen tietueen nykyiseksi tietueeksi.

Tallenna projekti -painike

7 Sulje Code-ikkuna ja tallenna muutokset napsauttamalla Tallenna projekti -painiketta.

Käynnistä ohjelma.

MyFindData-ohjelman suorittaminen

Käynnistä-painike

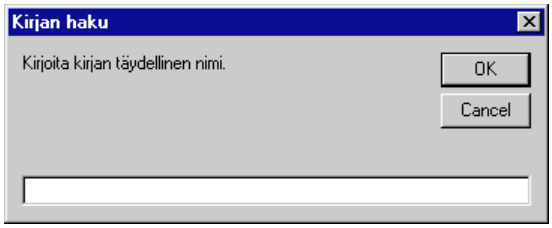
Valmis FindData -ohjelma on kansiossa \VB5Sbs\Less11.

1 Napsauta työkalurivin Käynnistä-painiketta.

Biblio.mdb-tietokannan Titles-taulusta luetut tiedot ilmestyvät tekstiruutuihin samanlaisina kuin edellä.

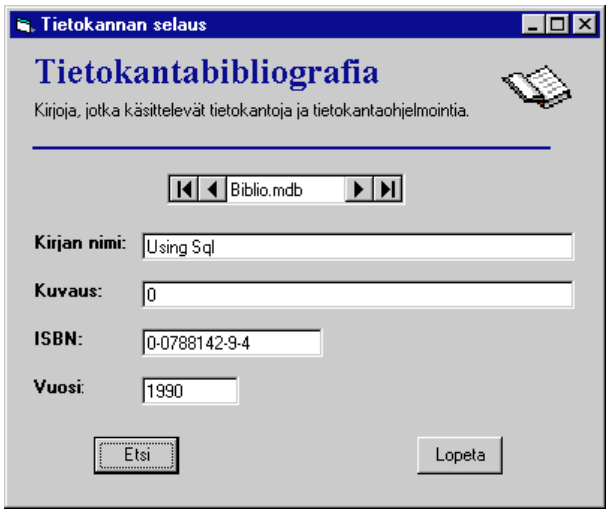
2 Napsauta Etsi-painiketta.

Kirjan haku -valintaikkuna avautuu seuraavan sivun kuvan mukaisena.



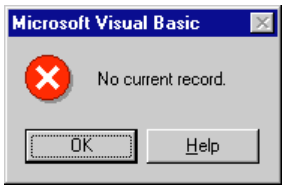
3 Kirjoita valintaikkunaan **Using SQL** ja paina ENTER.

cmdFind_Click-aliohjelma etsii tätä tekstiä tietokannan Title-kentästä ja pysähtyy seuraavaan tietueeseen:



4 Napsauta Find-painiketta uudelleen, kirjoita **Tietokannat** ja paina ENTER.

Aliohjelma ei löydä *Tietokannat*-nimistä kirjaa, joten se näyttää tällaisen ilmoituksen:



5 Sulje ilmoitusikkuna napsauttamalla OK.

MoveFirst-metodi valitsee tietokannan ensimmäisen tietueen nykyiseksi tietueeksi. Koska Biblio.mdb on indeksoitu ISBN-kentän mukaan ja nykyinen tietue valitaan Title-kentän mukaan, ohjelma näyttää aakkosjärjestyksessä ensimmäisen tietueen.

6 Lopeta ohjelma napsauttamalla Lopeta-painiketta.

Ohjelman suoritus päättyy, ja ohjelmointiympäristö palaa näkyviin.

Tietueiden lisääminen Biblio.mdb-tietokantaan

Kun lisäät uusia tietueita tietokantaan, määritä dataobjektin ReadOnly-ominaisuudeksi False ja kutsu tapahtuma-aliohjelmassa AddNew-metodia. Kun uusi, tyhjä tietue näkyy lomakkeessa, käyttäjä voi täyttää tarpeelliset kentät ja siirtyä sitten johonkin toiseen tietueeseen napsauttamalla dataobjektin jotakin painiketta. Kun käyttäjä siirtyy toiseen tietueeseen, uusi tietue sijoittuu tietokantaan aakkosjärjestyksen mukaiselle paikalleen.

Seuraavasta ohjelmasta opit, miten ReadOnly-ominaisuutta ja AddNew-metodia käytetään lisättäessä uusia tietueita tietokantaan. InputBox-funktio antaa käyttäjälle visuaalista palautetta.

Tietueiden lisääminen tietokantaan

Ennen kuin alat muokata ohjelmaa, tallenna se uudella nimellä.

1 Valitse File-valikosta komento Save MyFindData.frm As ja tallenna MyFindData-lomake nimellä **MyAddData.frm**. Valitse komento Save Project As ja tallenna projekti nimellä **MyAddData.vbp**.

2 Napsauta lomakkeen datBiblio-objektia (dataobjektia) ja avaa Properties-ikkuna.

3 Määritä datBiblio-objektin ReadOnly-ominaisuudeksi False.

ReadOnly-ominaisuus määrää, miten tietokanta avataan. Kun se on False, käyttäjä voi tehdä muutoksia ja lisätä uusia tietueita tietokantaan.

4 Napsauta Komentopainike-työkalua ja piirrä lomakkeeseen komentopainike Etsi-painikkeen oikealle puolelle.

Komentopainike-työkalu

5 Määritä komentopainikeobjektin ominaisuudet näin:

Objekti	Ominaisuus	Asetus
Command1	Caption	”Lisää”
	Name	cmdAdd

Lomakkeen pitäisi näyttää seuraavan kuvan kaltaiselta:

6 Kaksoisnapsauta Lisää-painiketta, jotta näet tapahtuma-aliohjelman cmdAdd_Click Code-ikkunassa.

7 Kirjoita tapahtuma-aliohjelmaan seuraavat ohjelmalauseet:

AddNew-metodi lisää tietokantaan uuden tietueen.

```
prompt$ = "Kirjoita uusi tietue ja napsauta vasenta
nuolipainiketta."
reply = MsgBox(prompt$, vbOKCancel, "Lisää  tietue")
If reply = vbOK Then           'jos käyttäjä napsauttaa OK
    txtTitle.SetFocus          'siirrä kohdistin Title-ruutuun
    datBiblio.Recordset.AddNew   'ja lue uusi tietue
    'set PubID field to 14 (Biblio.mdb  vaatii
    datBiblio.Recordset.PubID = 14 'tämän  kentän)
End If
```

Aliohjelma näyttää ensin valintaikkunan, jossa on tietojen lisäysohjeet käyttäjälle. MsgBox-funktiossa on argumenttina vbOKCancel (Visual Basicin määrittämä lukuarvoinen vakio), joka tarkoittaa, että valintaikkunassa on OK- ja Peruuta (Cancel) -painikkeet. Jos käyttäjä napsauttaa OK, AddNew-metodi luo tietueen. Jos käyttäjä napsauttaa Peruuta, tehtävä keskeytyy. Tapahtuma-aliohjelma myös siirtää kohdistimen Title-tekstiruutuun kutsumalla SetFocus-metodia. SetFocus-metodilla voidaan aktivoida mikä tahansa objekti, joka on aktivoitavissa.

If-rakenteen viimeinen lause määrittää Biblio.mdb-tietokannan PubID-kentän arvon. Tämä kenttä tarvitaan jokaisessa uudessa tietueessa, ja koska selausohjelmassa ei ollut sitä, sinun on lisättävä se itse. Arvo 14 merkitsee, että kirjan julkaisija on Microsoft Press.

Tallenna projekti -painike

8 Sulje Code-ikkuna ja tallenna muutokset napsauttamalla Tallenna projekti -painiketta.

Lisää seuraavaksi tietokantaan uusi tietue napsauttamalla Add-painiketta.

MyAddData-ohjelman suorittaminen

Käynnistä-painike

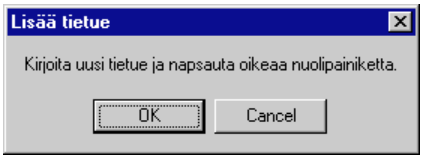
Valmis AddData-ohjelma on kansiossa \Vb5Sbs\Less11.

1 Napsauta työkalurivin Käynnistä-painiketta.

Tekstiruuduissa näkyvät Biblio.mdb-tietokannan Titles-taulsta luetut tiedot.

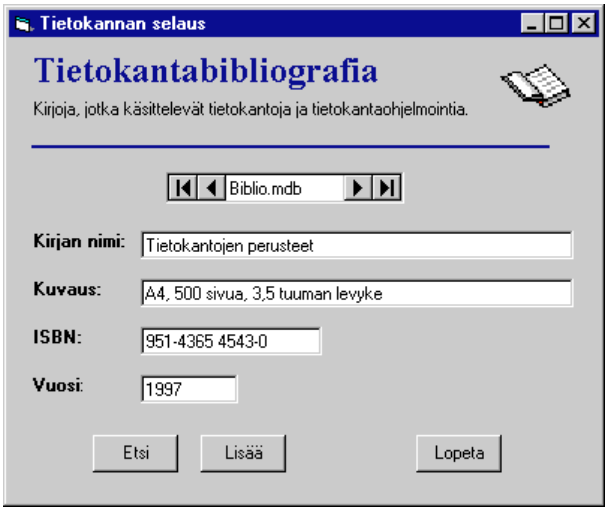
2 Napsauta Lisää-painiketta.

Lisää tietue -valintaikkuna avautuu seuraavan kuvan mukaisena:



3 Napsauta OK-painiketta.

Lomakkeeseen ilmestyy uusi, tyhjä tietue. Kirjoita siihen seuraavan kuvan mukaiset tiedot. Siirry kentästä toiseen SARKAINNÄPPÄIMELLÄ.



4. Kun olet kirjoittanut tietueen tiedot, napsauta dataobjektin ulompaa vasemmalle osoittavaa nuolipainiketta.

   Ohjelma lisää viimeiseksi tietueeksi kirjan *Tietokantojen perusteet* tiedot ja näyttää sitten tietokannan ensimmäisen tietueen.

5. Napsauta Etsi-painiketta, kirjoita **Tietokantojen perusteet** ja paina ENTER.

   Kirjan *Tietokantojen perusteet* tietue näkyy lomakkeessa.

6. Napsauta dataobjektin sisempää vasemmalle osoittavaa nuolipainiketta.

   Tietokannan edellinen tietue tulee näkyviin. Voit lisätä tietokantaan miten monta tietuetta tahansa napsauttamalla Lisää-painiketta.

7. Lopeta ohjelma napsautamalla Lopeta-painiketta.

   Ohjelman suoritus päättyy, ja ohjelmointiympäristö palaa näkyviin.

Tietueiden poistaminen Biblio.mdb-tietokannasta

Kun haluat poistaa tietueen tietokannasta, tuo se näkyviin ja poista se käyttäen Delete-metodia ja Recordset-objektia. Ennen tietokannan avaamista sinun on määritettävä dataobjektin ReadOnly-ominaisuudeksi False. (Näin teit myös aiemmin AddNew-metodin yhteydessä.) Tietueen poistamisen jälkeen sinun pitäisi tuoda näkyviin jokin toinen tietue, koska dataobjekti ei automaattisesti siirry toiseen tietueeseen. Yleensä on parasta näyttää tietokannan ensimmäinen tietue MoveFirst-metodilla.

Seuraavassa esimerkissä näet, miten Visual Basicilla poistetaan tietueita Biblio.mdb-tietokannasta. Kiinnitä erityistä huomiota MsgBox-funktioon ohjelmassa. Koska dataobjektissa ei ole kumoamistoimintoa, on tärkeää, että ohjelma pyytää komennolle vahvistuksen ennen kuin se tuhoaa tietueen.

VAROITUS Delete-metodi poistaa tietueen pysyvästi tietokannasta. Älä anna käyttäjille mahdollisuutta kutsua tätä metodia, paitsi jos heillä pitää olla oikeus poistaa tietueita.

Tietueiden poistaminen tietokannasta

Ennen kuin alat muokata MyAddData-ohjelmaa, tallenna se toisella nimellä, jotta alkuperäinen ohjelma säilyy.

1. Valitse File-valikosta komento Save MyAddData.frm As ja tallenna MyAddData-lomake nimellä **MyDelData.frm**. Valitse komento Save Project As command ja tallenna projekti nimellä **MyDelData.vbp**.
2. Napsauta lomakkeen datBiblio-objektia (dataobjektia) ja avaa Properties-ikkuna.

Komentopainike-työkalu

3 Varmista, että datBiblio-objektin ReadOnly-ominaisuus on False.

4 Napsauta Komentopainike-työkalua ja piirrä lomakkeeseen komentopainike Lisää-painikkeen oikealle puolelle.

5 Määritä komentopainikeobjektin ominaisuudet näin:

Objekti	Ominaisuus	Asetus
Command1	Caption	”Poista”
	Name	cmdDelete

Lomakkeen pitäisi olla seuraavan kuvan mukainen:

6 Kaksoisnapsauta Poista-painiketta, jotta tapahtuma-aliohjelma cmd-Delete_Click ilmestyy Code-ikkunaan.

7 Kirjoita aliohjelmaan seuraavat ohjelmalauseet:

Delete-metodi poistaa tietueen tietokannasta.

```
prompt$ = "Haluatko varmasti poistaa tämän tietueen?"
reply = MsgBox(prompt$, vbOKCancel, "Poista tietue")
If reply = vbOK Then                'jos käyttäjä napsauttaa OK
    datBiblio.Recordset.Delete      'poista nykyinen tietue
    datBiblio.Recordset.MoveNext    'siirry seur. tietueeseen
End If
```

Aliohjelma kysyy ensin valintaikkunassa, haluaako käyttäjä poistaa nykyisen tietueen. MsgBox-funktiossa on jälleen argumenttina vbOKCancel, jotta käyttäjä voi peruuttaa komennon. Jos käyttäjä napsauttaa OK, aliohjelma poistaa nykyisen tietueen Delete-metodilla ja näyttää seuraavan tietueen MoveNext-metodilla. Jos käyttäjä napsauttaa Peruuta (Cancel), tietuetta ei poisteta.

Tallenna projekti -painike

8 Sulje Code-ikkuna ja tallenna muutokset napsauttamalla Tallenna projekti -painiketta.

Nyt voit poistaa kirjan *Tietokantojen perusteet* tietueen tietokannasta napsauttamalla Poista-painiketta.

MyDelData-ohjelman suorittaminen

Käynnistä-painike

Valmis DelData-ohjelma on kansiossa \Vb5Sbs\Less11.

1 Napsauta työkalurivin Käynnistä-painiketta.

Biblio.mdb-tietokannan Titles-taulusta luetut tiedot ilmestyvät tekstiruutuihin.

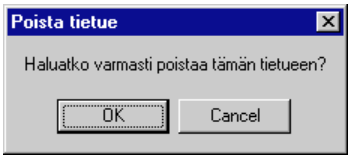
2 Tuo näkyviin kirjan *Tietokantojen perusteet* tietue käyttämällä Etsi-painiketta.

Ohjelma näyttää edellisessä tehtävässä lisäämäsi tietueen.

VAROITUS Seuraavissa vaiheissa poistetaan tietue pysyvästi Biblio.mdb-tietokannasta.

3 Napsauta lomakkeen Poista-painiketta.

Näyttöön tulee seuraavan kuvan mukainen Poista tietue -valintaikkuna:



4 Poista tietue napsauttamalla OK-painiketta.

Kirjan *Tietokantojen perusteet* tietue katoaa tietokannasta.

5 Lopeta ohjelma napsauttamalla Lopeta-painiketta.

Nyt osaat dataobjektin käyttämisen perusteet. Jos haluat tietää lisää tietokantasovellusten kirjoittamisesta, etsi Visual Basicin ohjeista hakusanaa *Recordset* ja tutustu Visual Basicin mukana toimitettuihin esimerkkisovelluksiin. Esimerkkisovellukset käsittelevät Biblio.mdb-tietokantaa, joten osaat tunnistaa monia niiden kenttiä ja tietueita.

Lisätietoja: Tiedoston varmistuskopiointi

Tietokannoissa säilyttämäsi tiedot ovat todennäköisesti hyvin tärkeitä, joten olet pienessä pulassa, jos niille tapahtuu jotakin. Siksi sinun pitäisi aina varmistuskopioida tietokanta ennen kuin alat muokata sitä. Jos jotakin menee vikaan kopiota muokatessasi, voit muitta mutkitta palata alkuperäiseen tietokantaan. Voit huolehtia kopioimisesta varmistusohjelmalla Windows Resurssienhallinnalla tai tietokantaohjelman erityisellä varmistustoiminnolla. Voit varmuuden vuoksi vielä tehdä kopion yhdestä tai useasta tiedostosta Visual Basic -ohjelmassa FileCopy-lauseella. Se tekee erillisen kopion tiedostosta samalla tavoin kuin Windows Resurssienhallinnan komento Muokkaa Kopioi (Edit Copy). Sen muoto on tällainen:

FileCopy-lause tekee tiedostosta kopion.

```
FileCopy lähdepolku kohdepolku
```

Lähdepolku on kopioitavan tiedoston polkunimi ja *kohdepolku* kohdetiedoston polkunimi.

HUOMAA FileCopy ei toimi, jos *lähdepolun* määrittämä tiedosto on avoinna.

Seuraava tehtäväsi on lisätä MyDelData-ohjelmaan kopiointitoiminto kirjoittamalla Form_Load-aliohjelmaan FileCopy-lause.

Biblio.mdb-tietokannan kopioiminen FileCopy-lauseella

Tallenna ensin MyDelData-ohjelma uudella nimellä, jotta alkuperäinen ohjelma säilyy. Ellei se ole enää avoinna, lataa DelData.vbp levyltä ja tuo sen lomake näkyviin.

1. Valitse File-valikosta komento Save MyDelData.frm As ja tallenna MyDelData-lomake nimellä **MyBackup.frm**. Valitse komento Save Project As ja tallenna projekti nimellä **MyBackup.vbp**.
2. Ellei lomake ole vielä näkyvissä, valitse se Project-ikkunasta ja napsauta Näytä objekti -painiketta. Kaksoisnapsauta lomaketta (ei objektia), jotta tapahtuma-aliohjelma Form_Load ilmestyy Code-ikkunaan.

   Kun sijoitat FileCopy-lauseen käynnistysaliohjelmaan Form_Load, käyttäjä voi kopioida tietokannan ennen kuin hän tekee siihen muutoksia.

3 Kirjoita Form_Load-aliohjelmaan seuraavat ohjelmalauseet:

```
prompt$ = _
      "Haluatko tehdä tietokannasta varmistuskopion?"
reply = MsgBox(prompt$, vbOKCancel, datBiblio.DatabaseName)
If reply = vbOK Then   'kopioi tietokanta, jos käyttäjä
                        napsauttaa OK
    FileNm$ = InputBox$ _
              ("Kirjoita varmistuskopion polkunimi.")
    If FileNm$ <> "" Then _
        FileCopy datBiblio.DatabaseName, FileNm$
End If
```

FileCopy-lause tekee tietokannasta kopion.

Aliohjelma kysyy ohjelman käynnistyttyä, haluaako käyttäjä tehdä tietokannasta varmistuskopion. MsgBox-funktiossa on argumenttina vbOKCancel, jotta käyttäjä voi valita, kopioidaanko tietokanta vai ei. Tällä kertaa MsgBox-funktion kutsussa on myös DatabaseName-ominaisuus, jolla näytetään tietokannan nimi valintaikkunan otsikkorivillä. Jos käyttäjä napsauttaa OK, ohjelma kysyy varmistuskopion polkunimen toisessa ilmoitusikkunassa ja sitten kopioi tiedoston FileCopy-lauseella.

Tallenna projekti -painike

4 Sulje Code-ikkuna ja tallenna muutokset napsauttamalla Tallenna projekti -painiketta.

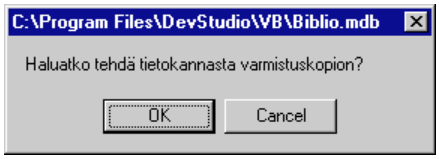
Käynnistä ohjelma, jotta näet, miten varmistuskopiointi toimii.

MyBackup-ohjelman suorittaminen

Käynnistä-painike

1 Napsauta työkalurivin Käynnistä-painiketta.

Näyttöön tulee seuraava valintaikkuna, jossa kysytään, haluatko tehdä tietokannasta kopion:



2 Hyväksy kopiointi napsauttamalla OK-painiketta.

Seuraavassa valintaikkunassa kysytään kopion polkunimi.

Valmis Backup-ohjelma on kansiossa \Vb5Sbs\Less11.

3 Kirjoita **c:\vb5sbs\less11\mybiblio.mdb** ja napsauta OK.

Visual Basic kopioi Biblio.mdb-tietokannan harjoituksen 11 hakemistoon nimellä Mybiblio.mdb. Jos teet virheen Biblio.mdb-tiedostoa muokatessasi, voit turvautua varmistuskopioon.

4 Lopeta ohjelma napsauttamalla Lopeta-painiketta.

Jos haluat jatkaa seuraavaan harjoitukseen

➤ Jätä Visual Basic käyntiin ja siirry harjoitukseen 12.

Jos haluat lopettaa Visual Basicin

➤ Valitse File-valikosta komento Exit.
Jos näyttöön tulee Save-valintaikkuna, napsauta Yes.

Harjoituksen tiivistelmä

Tehtävä	Ohje
Avata tekstitiedosto	Käytä Open...For Input -lausetta. Esimerkki: `Open CmnDialog1.FileName For Input As #1`
Lukea tekstitiedostosta yksi rivi	Käytä Line Input -lausetta. Esimerkki: `Line Input #1, LineOfText$`

Tarkistaa, onko tiedosto lopussa	Käytä EOF-funktiota. Esimerkki:
	`Do Until EOF(1)`
	`  Line Input #1, LineOfText$`
	`  Text$ = Text$ & LineOfText$ & Wrap$`
	`Loop`

Sulkea avoin tiedosto	Käytä Close-lausetta. Esimerkki: `Close #1`

Näyttää tekstitiedoston sisältö	Kopioi teksti avoimesta tiedostosta merkkijonomuuttujaan Line Input -lauseella ja sijoita merkkijonomuuttuja tekstiruutuobjektiin. Esimerkki:
	`Do Until EOF(1)`
	`  Line Input #1, LineOfText$`
	`  Text$ = Text$ & LineOfText$ & Wrap$`
	`Loop`
	`txtDisplay.Text = Text$`

Näyttää Avaa-valintaikkuna (Open)	Käytä yleisen valintaikkunaobjektin ShowOpen-metodia. Esimerkki: `CmnDialog1.ShowOpen`
Luoda tekstitiedosto	Käytä Open...For Output -lausetta. Esimerkki: `Open CmnDialog1.FileName For Output As #1`

Tehtävä	Ohje
Näyttää Tallenna nimellä -valintaikkuna (Save As)	Käytä yleisen valintaikkunaobjektin ShowSave-metodia. Esimerkki: `CmnDialog1.ShowSave`
Tallentaa tekstiä tiedostoon	Käytä Print # -lausetta. Esimerkki: `Print #1, txtNote.Text`
Avata tietokanta	Lisää lomakkeeseen dataobjekti Data-työkalulla ja sijoita sen DatabaseName-ominaisuuteen tietokannan nimi. Määritä tietokannan tyyppi Connect-ominaisuudella ja tietueen tyyppi RecordsetType-ominaisuudella.
Avata tietokanta vain lukemista varten	Määritä dataobjektin ReadOnly-ominaisuudeksi True.
Näyttää tietueen kenttä tekstiruutuobjektissa	Määritä tekstiruudun DataField- ja DataSource-ominaisuudet.

Etsiä tietoja tietokannasta	Kysy käyttäjältä etsittävä merkkijono ja käytä Recordset-objektin Index-, Seek-, NoMatch- ja MoveFirst-ominaisuuksia tapahtuma-aliohjelmassa. Esimerkki:
	`Prompt$ = "Anna kirjan koko nimi."`
	`SearchStr$ = InputBox(Prompt$, "Kirjahaku")`
	`datBiblio.Recordset.Index = "Nimi"`
	`datBiblio.Recordset.Seek "=", SearchStr$`
	`If datBiblio.Recordset.NoMatch Then`
	`MsgBox ("Kirjaa ei löytynyt.")`
	`End If`

Lisätä tietokantaan tietue	Käytä Recordset-objektin AddNew-metodia. Esimerkki: `datBiblio.Recordset.AddNew`
Lisätä tietokannan tietueeseen kenttä	Määritä kentän nimi Recordset-objektin ominaisuutena. Varmista, että tietojen muoto on oikea. Esimerkki: `datBiblio.Recordset.PubID = 14`
Poistaa tietokannasta tietue	Käytä Recordset-objektin Delete-metodia. Esimerkki: `datBiblio.Recordset.Delete`

Tehtävä	Ohje
Näyttää tietokannan ensimmäinen tietue	Käytä Recordset-objektin MoveFirst-metodia. Esimerkki: `datBiblio.Recordset.MoveFirst`
Kopioida tiedosto	Käytä FileCopy-lausetta. Esimerkki: `FileCopy datBiblio.DatabaseName, FileNm$`
Aktivoida objekti (siirtää kohdistin siihen)	Käytä objektin SetFocus-metodia. Esimerkki: `txtTitle.SetFocus`

Kun tarvitset tietoja aiheesta	**Valitse Visual Basicin Help-valikosta Microsoft Visual Basic Help Topics, valitse Index-välilehti ja...**
Tekstitiedostojen käsittely	Kirjoita **files, input and output**.
Tietokantojen käsittely	Kirjoita **data control**.
Recordset-ominaisuudet ja -metodit	Kirjoita **Recordset properties**.

Tietoja seuraavasta harjoituksesta

Seuraavassa harjoituksessa opit käynnistämään Windows-sovelluksia Visual Basicista, ohjelmoimaan sovellusobjekteja käyttämällä Automation-toimintoa ja sisällyttämään Microsoft Wordin ja Microsoft Excelin kaltaisia sovellusobjekteja ohjelmiisi Object Browserilla.

Harjoitus

12

Yhteydet Microsoft Officeen

Arvioitu aika
45 min.

Tässä harjoituksessa opit

- muodostamaan yrityksen tietojärjestelmän käyttämällä sovellusobjekteja
- muodostamaan aktiivisia linkkejä tiedostoihin, jotka on tehty Windows-sovelluksilla
- tutkimaan sovellusobjekteja Object Browserilla
- ohjaamaan Microsoft Wordia Automation-toiminnolla (aiemmalta nimeltään OLE Automation).

Harjoituksessa 2 opit käynnistämään ohjelmasta Windows-sovelluksia OLE-työkalulla. Jos aihe on sinusta kiinnostava, tästä harjoituksesta on sinulle paljon hyötyä. Visual Basic on erityisesti suunniteltu ”liimaksi”, joka liittää toisiinsa erilaisten Windows-sovellusten tietoja ja toimintoja nopeasti ja tehokkaasti.

Tässä harjoituksessa opit laatimaan yrityksen tietojärjestelmän (EIS, Enterprise Information System) käyttämällä sovellus- eli OLE-objektia ja useiden Microsoft Office -sovellusten tietoja. Opit myös tutkimaan Windows-sovellusten sovellusobjekteja Object Browserilla sekä sisällyttämään projekteihisi Microsoft Wordin ominaisuuksia Automation-toiminnolla.

Yrityksen tietojärjestelmän luominen

Yrityksen tietojärjestelmä on loppukäyttäjälle tarkoitettu sovellus, joka antaa tärkeitä operatiivisia tietoja ja jossa on johdonmukainen ja kätevä käyttöliittymä. Sen laatii tavallisesti yrityksen atk-osasto tai tietokanta-asiantuntija, koska nämä

henkilot ovat tottuneet hakemaan ja muotoilemaan tietoja, jotka ovat peräisin erityisistä tietokantaohjelmista, kuten Oraclesta tai Microsoft SQL Serveristä. Tietojärjestelmäsovellusten käyttäjät ovat yleensä työntekijöitä, joilla on hyvin vähän tai ei lainkaan kokemusta tietokannan hallinnasta — he ovat yrityksen päätöksentekijöitä, joiden pitää vaivatta löytää organisaatiota koskevat tiedot.

Tässä jaksossa teet yksinkertaisen tietojärjestelmän, joka näyttää Microsoft Access -tietokannan tietueita, Paintilla muokattuja työntekijöiden valokuvia sekä Microsoft Excelin taulukosta ja kaaviosta saatuja myyntitietoja. Ohjelma luodaan Visual Basicin OLE-objektista ja Microsoft Office -sovelluksista.

Yrityksen tietojärjestelmien käyttökohteita

Hyvässä yrityksen tietojärjestelmässä on yhdistetty usean sovelluksen tiedot ja ominaisuudet tehokkaaksi liikkeenjohdon apuvälineeksi. Laadin seuraavan luettelon tietojärjestelmäsovelluksista, jotta voit miettiä, miten voisit muodostaa Visual Basicin sovellusobjekteista ja Office-asiakirjoista mukautettuja ratkaisuja yrityksesi tai organisaatiosi käyttöön. Kuvaan useita tietojärjestelmän rakentamisessa tarvittavia menetelmiä tässä harjoituksessa.

Tilausten käsittely Käyttäjät voivat katsella tietojen syöttölomakkeita ja saapuvia tilauksia Excelillä ja laatia myyntiraportteja Wordilla.

Henkilöstönhallinta Käyttäjät voivat käsitellä työntekijöiden tietueita Accessilla, tutkia Word-asiakirjoihin tallennettuja työntekijöitä koskevia arvioita sekä muokata valokuvia Paintilla.

Taloudellinen analysointi Käyttäjät voivat tutkia Excel-tiedostoihin tallennettuja kirjanpito- ja tilastotietoja sekä verkosta saatuja investointitietoja.

Varastonvalvonta Myyntihenkilöstö voi tarkastella varastotietoja, jotka on tallennettu Oracle- tai SQL Server- tietokantoihin, pystyäkseen tekemään valmistukseen ja hinnoitteluun liittyviä päätöksiä.

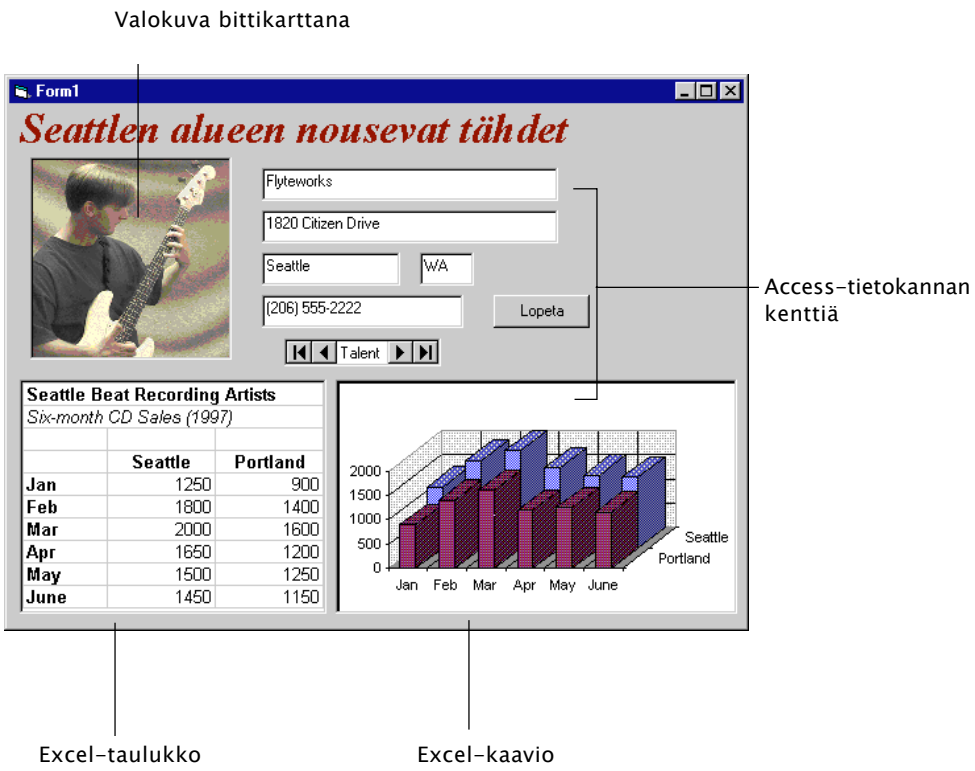
Johdon tietojärjestelmä Ylin johto voi lähettää sähköpostia Microsoft Maililla, lukea Word-asiakirjoihin tallennettuja yrityksen tiedotteita sekä suorittaa Accessilla mukautettuja tietokantakyselyjä.

Projektinhallinta Käyttäjät voivat hallita aikataulutietoja Microsoft Projectilla, seurata markkinaennusteita Excelillä, laatia mukautettuja tilanneraportteja Wordilla sekä valmistella elektronisia esityksiä Microsoft PowerPointilla.

OLE-työkalun käyttö

OLE-työkalulla (toiselta nimeltään OLE-säilötyökalulla) lisätään Visual Basic -ohjelmaan sovellus- eli OLE-objekteja. Käytettävissä olevat objektit riippuvat siitä, mitä Windows-sovelluksia järjestelmään on asennettu. Jokainen objekteja tukeva sovellus ja sen tuntemat objektit on määritetty Windowsin rekisterissä (järjestelmätietokannassa, johon Windows tallentaa sovelluksia koskevia tietoja). Valitse objektit Insert Object -valintaikkunasta, joka avautuu, kun ensimmäisen kerran käytät OLE-työkalua. Esimerkkejä sovellusobjekteista ovat Excel-taulukot, Excel-kaaviot, Word-asiakirjat ja Microsoft ClipArt -kuvat. Sovellusobjektit voivat olla uusia, tyhjiä asiakirjoja tai levyltä ladattuja tiedostoja.

Seuraavassa ohjelmassa käytät dataobjektia, kuvaobjektia, kahta OLE-objektia ja useita teksti- ja tekstiruutuobjekteja tehdäksesi sovelluksen, jolla tallennetaan tiedot Seattlen alueen levyttävistä artisteista. Tämä mallisovellus näyttää valokuvan, Access-tietokannasta saatuja tietueita sekä Excel-taulukon ja -kaavion sisällön. Valmiin sovelluksen ikkuna näyttää tällaiselta:



MusicEIS-sovelluksen muodostaminen

1. Käynnistä Visual Basic.

   Jos Visual Basic on jo käynnissä, valitse File-valikosta komento New Project. Sovelluksesta tulee suurempi kuin oletuslomake, joten sinun pitää suurentaa lomaketta.

2. Suurenna lomakkeen korkeutta ja leveyttä vetämällä Project1-ikkunan ja Form-ikkunan oikeaa alakulmaa alaspäin ja oikealle. Sinun pitää ehkä sulkea tai pienentää ohjelmointityökaluja, jotta näyttöön tulee tarpeeksi tilaa.

   Kun muutat lomakkeen tai ikkunan kokoa, hiiren osoitin näyttää tällaiselta:

3. Napsauta Teksti-työkalua ja piirrä pitkä tekstiobjekti lomakkeen yläosan keskelle.

   Teksti-työkalu

   Tähän objektiin sijoitetaan ohjelman otsikko. Katso edellisen sivun kuvasta objektien koot ja paikat.

4. Napsauta Kuva-työkalua ja piirrä kuvaobjekti lomakkeen vasempaan reunaan, tekstiobjektin alapuolelle.

   Kuva-työkalu

   Tällä objektilla näytetään valokuva, joka on skannattu ja tallennettu .bmp-tiedostoon.

5. Napsauta Tekstiruutu-työkalua ja piirrä kuvaobjektin oikealle puolelle viisi tekstiruutuobjektia.

   Tekstiruutu-työkalu

   Nämä tekstiruudut kytketään Access-tietokantaan, joka sisältää kuvitteellisten muusikoiden nimiä, osoitteita ja puhelinnumeroita.

6. Napsauta Data-työkalua ja lisää dataobjekti alimman tekstiruutuobjektin alapuolelle

   Data-työkalu

   Dataobjektia käytetään muusikkotietokannan tietueiden selaamiseen.

7. Napsauta Komentopainike-työkalua ja piirrä komentopainike alimman tekstiruutuobjektin oikealle puolelle.

   Komentopainike-työkalu

   Tällä komentopainikkeella lopetetaan ohjelma.

8. Määritä lomakkeen objektien ominaisuudet seuraavan taulukon mukaisesti. Löydät muusikkotietokannan ja skannatun .bmp-tiedoston kansiosta \Vb5Sbs\Less12.

Objekti	Ominaisuus	Asetus
Form1	Caption	"Kykytietokanta"
Label1	Caption	"Seattlen alueen nousevat tähdet
	Font	Times New Roman, Bold Italic, 24 pistettä
	ForeColor	Dark red
Image1	BorderStyle	1 - Fixed Single
	Stretch	True
	Picture	C:\Vb5Sbs\Less12\Flytwork.bmp
Data1	Caption	"Talent"
	Connect	Access
	DatabaseName	C:\Vb5Sbs\Less12\Talent.mdb
	ReadOnly	True
	RecordSource	Artists
Text1	DataSource	Data1
	DataField	Name
	Text	(Tyhjä)
Text2	DataSource	Data1
	DataField	Address
	Text	(Tyhjä)
Text3	DataSource	Data1
	DataField	City
	Text	(Tyhjä)
Text4	DataSource	Data1
	DataField	State
	Text	(Tyhjä)
Text5	DataSource	Data1
	DataField	WorkPhone
	Text	(Tyhjä)
Command1	Caption	"Lopeta"

9 Kaksoisnapsauta Quit-painiketta ja kirjoita tapahtuma-aliohjelmaan Command1_Click lause **End**.

Tämä lause lopettaa sovelluksen suorituksen, kun Lopeta-painiketta napsautetaan.

10 Valitse komento Save From1 As ja tallenna lomake nimellä **MyMusicEIS.frm**. Valitse komento Save Project As ja tallenna projekti nimellä **MyMusicEIS.vbp**. Sijoita tiedostot kansioon Less12.

Nyt voit sisällyttää sovellukseen Excel-taulukon ja -kaavion käyttämällä OLE-työkalua.

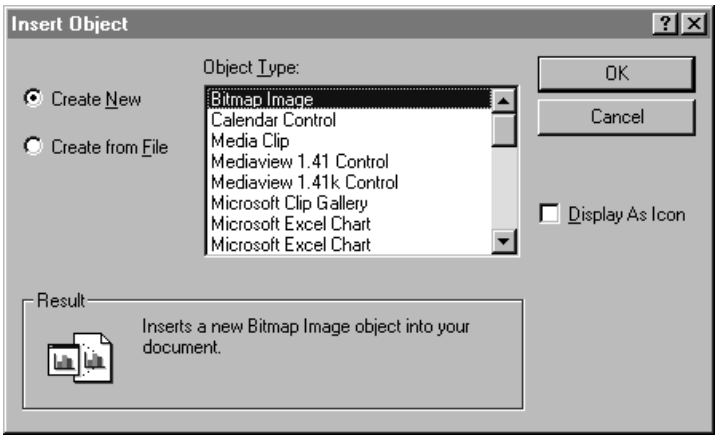
OLE-objektien lisääminen

Seuraavissa vaiheissa kytket ohjelmasi Excel-taulukkoon ja -kaavioon OLE-työkalulla.

OLE-työkalu

1. Napsauta OLE-työkalua ja piirrä lomakkeen vasempaan alakulmaan suuri suorakulmio.

   Kun olet piirtänyt suorakulmion ja vapautat hiiren painikkeen, Insert Object -valintaikkuna avautuu seuraavan kuvan kaltaisena:



Insert Object -valintaikkunassa näkyy objektit, jotka on tallennettu järjestelmän rekisteriin.

   Valintaikkunassa on lueteltu objektit, joita voit linkittää tai upottaa ohjelmaasi. *Linkitetty objekti* sisältää tietoja, joita käsitellään lähdesovelluksella, jolla ne on luotu, ja jotka säilytetään lähdesovelluksen tiedostossa, erillään Visual Basic -ohjelmasta. *Upotettu objekti* sisältää tietoja, jotka on tallennettu Visual Basic -ohjelman sisään. Muut sovellukset voivat käyttää linkitettyjä objekteja — esimerkiksi työtoverisi voivat muokata Visual Basic -ohjelmaasi linkitettyjä Excel-myyntitaulukoita — mutta upotettu objekti on ainoastaan yhden sovelluksen (tässä tapauksessa MyMusicEIS-ohjelman) käytettävissä.

2. Napsauta Insert Object -valintaikkunan Create From File -valintanappia.

   Objektin polkunimi ilmestyy valintaikkunaan. Valitse Create From File, kun haluat lisätä ohjelmaasi olemassa olevan tiedoston.

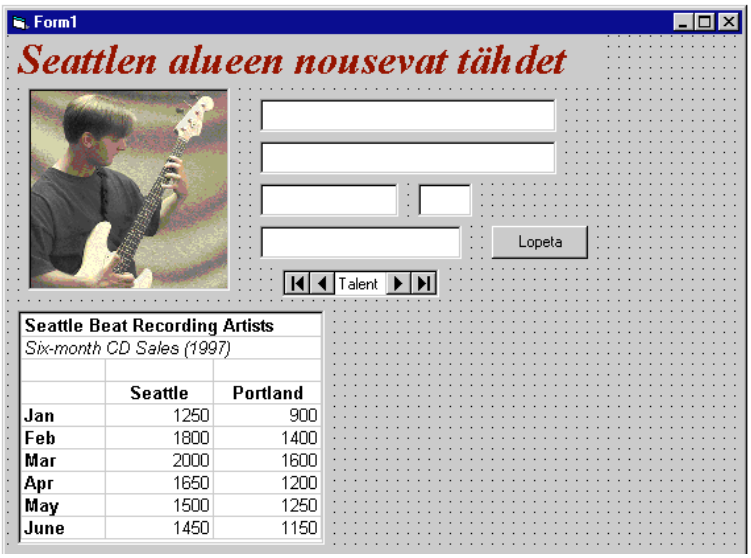
3. Muodosta linkitetty objekti valitsemalla Link-valintaruutu.

   Kun Link-valintaruutu on valittu, valitsemasi tiedoston kuva latautuu sovellusobjektiin. Koska tiedosto on linkitetty objekti, se on edelleen olemassa Visual Basic -ohjelmasi ulkopuolella, ja siihen tehdyt muutokset vaikuttavat ohjelmaasi.

4. Napsauta Browse-painiketta ja etsi Excel-tiedosto, joka on tarkoitus linkittää ohjelmaan.

5 Siirry kansioon Less12, napsauta Excel-taulukkoa Sales_97 ja napsauta Insert. Kun Insert Object -valintaikkuna jälleen avautuu, muodosta linkki Sales_97-tiedostoon napsauttamalla OK.

Excel-taulukon kuva ilmestyy hetken kuluttua sovellusobjektin sisälle. (Ellei taulukon koko sisältö näy, suurenna OLE-objektia.) Lomakkeen pitäisi näyttää tällaiselta:



Taulukko kertoo, kuinka monta paikallisten artistien tekemää CD-levyä on myyty puolen vuoden aikana. Koska OLE-objekti on linkitetty Excel-taulukkoon, taulukkoon Sales_97.xls tehdyt muutokset näkyvät ohjelmassasi, kun käynnistät sen.

Kopioi seuraavaksi myyntilukuja kuvaava kaavio Sales_97-taulukosta ja liitä se uuteen sovellusobjektiin.

HUOMAA Seuraavissa vaiheissa edellytetään, että järjestelmääsi on asennettu Microsoft Excel for Windows versio 5.0 tai uudempi (kuvissa näkyy Microsoft Excel 97).

Myyntikaavion kopioiminen ja liittäminen

OLE-työkalu

1 Napsauta OLE-työkalua ja piirrä suorakulmio lomakkeen oikeaan alakulmaan.

Kun vapautat hiiren painikkeen, Insert Object -valintaikkuna avautuu.

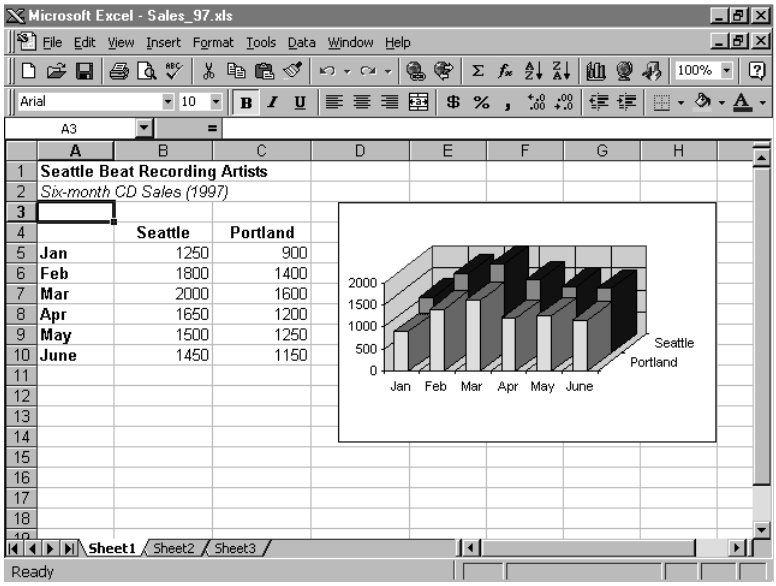
Kun haluat sijoittaa tiedoston osan OLE-objektiin, käytä komentoa Paste Special.

2 Sulje Insert Object -valintaikkuna napsauttamalla Cancel.

Tällä kertaa sovellusobjektiin on tarkoitus sijoittaa vain osa taulukkotiedostosta: Excel-kaavio. Kun tarvitset tietyn osan tiedostosta, sinun täytyy kopioida se lähdesovelluksesta ja liittää se sovellusobjektiin komennolla Liitä määräten (Paste Special).

3 Käynnistä Excel ja avaa taulukko Sales_97.xls, joka on kansiossa \Vb5Sbs\Less12.

Näytön pitäisi olla tällainen:



4 Napsauta myyntikaavion reunaviivaa ja valitse Muokkaa-valikosta (Edit) komento Kopioi (Copy).

Myyntikaavion kopio tallentuu Windowsin Leikepöydälle.

5 Palaa Visual Basic -ohjelmointiympäristöön.

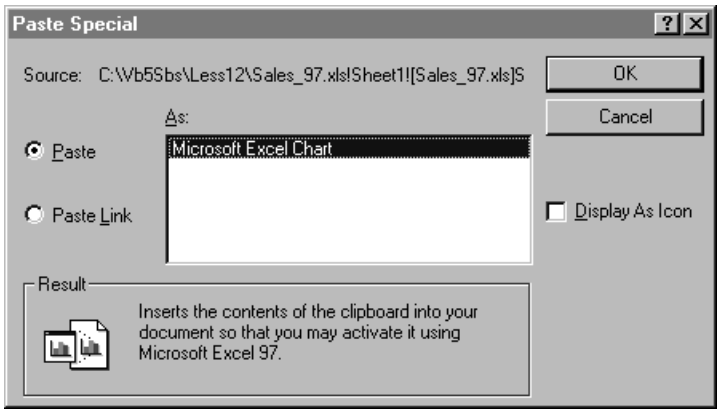
Voit valita Paste Special -komennon napsauttamalla hiiren oikeaa painiketta.

6 Siirrä hiiren osoitin tyhjän sovellusobjektin päälle ja napsauta hiiren oikeaa painiketta.

Sovellusobjektin viereen avautuu tilannekohtainen ponnahdusvalikko.

7 Valitse ponnahdusvalikosta komento Paste Special, jotta Excel-kaavio sijoittuu sovellusobjektiin.

Paste Special -valintaikkuna avautuu seuraavan kuvan mukaisena.

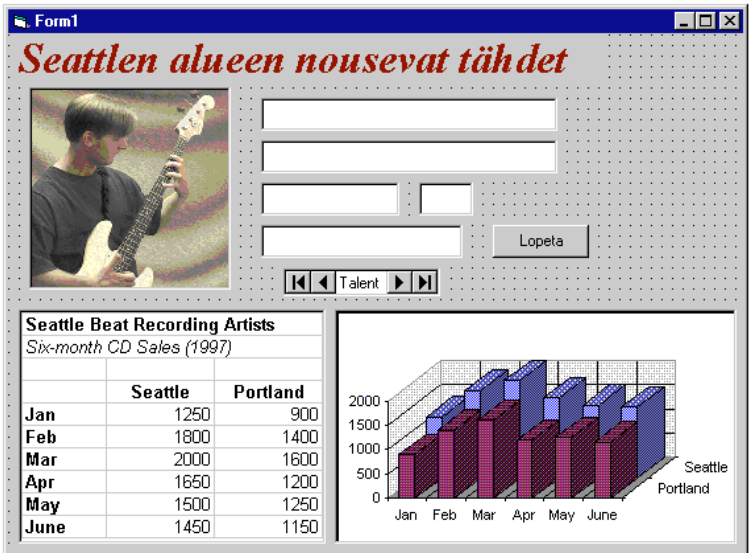


8 Napsauta Paste Link -valintanappia osoitukseksi siitä, että haluat liittää objektiin linkin, joka kohdistuu Excel-kaavioon.

Kun kaavio on linkitetty MyMusicEIS-ohjelmaan, siihen Excelillä tehdyt muutokset näkyvät ohjelmassa automaattisesti.

9 Linkitä kaavio napsauttamalla OK.

Myyntikaavio ilmestyy sovellusobjektin sisäpuolelle, kuten seuraavasta kuvasta näkyy:



Viimeinen vaihe on lisätä kaksi ohjelmalausetta, jotka päivittävät linkit Excel-tiedostoihin, kun ohjelma käynnistetään.

10 Kaksoisnapsauta lomaketta, jotta tapahtuma-aliohjelma Form_Load ilmestyy Code-ikkunaan, ja kirjoita aliohjelmaan tällaiset lauseet:

```
OLE1.Update
OLE2.Update
```

Lauseet päivittävät Visual Basic -ohjelmasi linkitettyihin Excel-tiedostoihin tehtyjen muutosten mukaisiksi Update-metodilla. Ne hidastavat ohjelman käynnistämistä hieman, mutta ne ovat välttämättömät, jos yrityksen muutkin työntekijät muokkaavat linkitettyjä tiedostoja.

Tallenna projekti –painike

11 Tallenna valmis ohjelma napsauttamalla Tallenna projekti -painiketta.

Käynnistä ohjelma, jotta näet, miten se esittää eri lähteistä saadut tiedot.

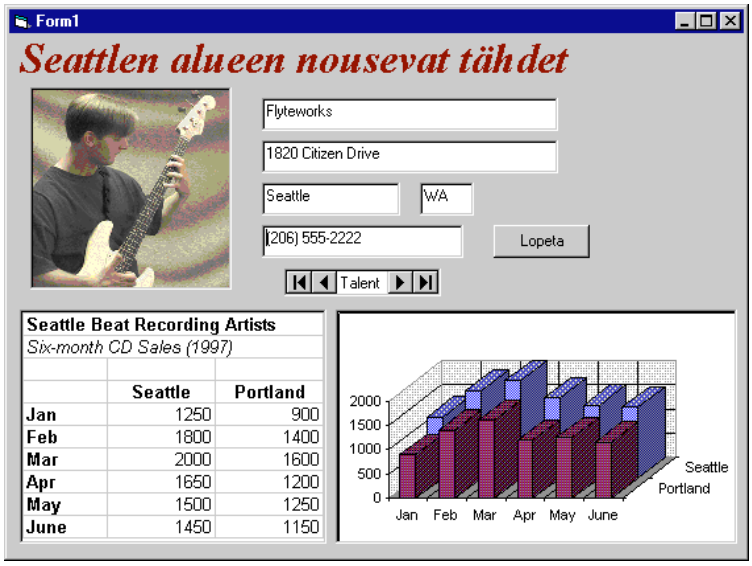
Tietojärjestelmäohjelman käynnistäminen

Käynnistä–painike

1 Käynnistä ohjelma napsauttamalla Käynnistä-painiketta.

Sovellusohjelma lukee tiedot kolmesta lähteestä ja esittää ne seuraavan kuvan mukaisesti:

Valmis MusicEIS–ohjelma on kansiossa \Vb5Sbs\Less12.



2 Selaa tietokantaa napsauttamalla dataobjektin sisempää oikealle osoittavaa nuolta.

Tekstiruutuihin ilmestyy eri nimiä, osoitteita ja puhelinnumeroita, kun napsautat dataobjektia. Esimerkissä on vain yksi valokuva, joten kuva ei vaihdu. Jos tietokannassa olisi valokuvakenttä, voisit näyttää kuva-kehyksessä joukon bittikarttoja määrittämällä kuvaobjektin DataField-ominaisuudeksi tietokannan valokuvakentän.

3. Kaksoisnapsauta lomakkeen taulukkoa.

   Visual Basic käynnistää Excelin eli sovelluksen, jolla taulukko on tehty, ja avaa taulukon Sales_97.xls. Koska lisäämäsi objekti on linkitetty taulukkoon, voit tehdä muutoksia alkuperäiseen taulukkoon ja tallentaa ne taulukkotiedostoon.

HUOMAA Jos et halua käyttäjien käynnistävän Exceliä ja tekevän muutoksia linkitettyyn tiedostoon, määritä OLE-objektin Enabled-ominaisuudeksi False.

4. Siirrä kohdistin taulukon soluun C7, kirjoita **0** (nolla) ja paina ENTER.

   Portlandissa maaliskuussa myytyjen CD-levyjen määrä on tämän jälkeen 0 eikä 1600.

5. Valitse Excelin Tiedosto-valikosta (File) Lopeta (Exit) ja tallenna muutokset vastaamalla Kyllä (Yes).

   Taulukkotiedosto päivittyy, ja Excelin suoritus päättyy. Ohjelmasi näyttää taulukossa uuden myyntiluvun. Sen sijaan kaavio *ei* ole muuttunut. Visual Basic päivittää sovellusobjektin sisällön vain, jos aktivoit linkin kaksoisnapsauttamalla objektia, jolloin linkitetty objekti avautuu, tai jos päivität linkin ohjelmallisesti Update-metodilla. Muuten Visual Basic säilyttää tiedot sellaisina kuin ne alkuaan ladattiin, jotta tiedot eivät ”katoa” kesken työn.

   Jos haluat Visual Basicin reagoivan objektien muutoksiin heti, voit kirjoittaa Update-metodin kutsun erityiseen OLE-tapahtuma-aliohjelmaan OLE1_Updated, joka suoritetaan aina sovellusobjektin muuttuessa. Kuten huomaat, Visual Basicilla on mahdollista määrätä, kuinka usein linkitetyt tiedot saatetaan ajan tasalle.

6. Lopeta ohjelman suoritus napsauttamalla Lopeta-painiketta.

OLE-objektien kuvaus loppuu tähän.

Sovellusobjektien ohjelmoiminen Automation-tekniikalla

Edellisessä jaksossa opit sisällyttämään Visual Basic -ohjelmiisi muilla Windows-sovelluksilla tehtyjä objekteja. Tässä jaksossa näet, miten voit lisätä Automation-tekniikalla ohjelmaasi Windows-sovellusten ominaisuuksia.
(Automation-tekniikka tunnettiin ohjelmointipiireissä aikaisemmin nimellä OLE Automation.)

Automation-tekniikkaa täydellisesti tukevat Windows-sovellukset luovuttavat toimintonsa muiden sovellusten käyttöön objektikokoelmana, johon kuuluu myös ominaisuuksia ja metodeja. Tällaisia Windows-sovelluksia kutsutaan *objekti-* tai *palvelinsovelluksiksi*. Niiden objekteja käyttäviä ohjelmia taas kutsutaan *ohjaaviksi* tai *asiakassovelluksiksi*. Automation-tekniikka on käytännön standardi, joka mahdollistaa kahden sovelluksen välisen viestinnän seuraavan kuvan mukaisella tavalla:

Automation-tekniikan käyttö Visual Basicissa

Visual Basic 5:llä voidaan tehdä sekä palvelin- että asiakassovelluksia, jotka tukevat Automation-tekniikkaa. Toimintojaan muille sovelluksille luovuttavien palvelinsovellusten tekeminen edellyttää kuitenkin Visual Basic Professional tai Enterprise Editionia ja on tämän kirjan aihepiirin ulkopuolella. Palvelinsovellusten toimintoja hyödyntävien asiakassovellusten tekeminen sen sijaan on yksinkertaista ja onnistuu Visual Basic 5:n kaikilla versioilla.

HUOMAA Kaikki Microsoft Office 97 -sovellukset (Excel, Word, Access, PowerPoint ja Microsoft Outlook) pystyvät luovuttamaan toimintojaan Automation-tekniikalla. Koska kunkin sovelluksen toiminnot ja objektit ovat erilaiset, sinun pitää tutustua tuotteen käyttöoppaisiin tai ohjeisiin osataksesi käyttää sovelluksen toimintoja omissa ohjelmissasi. Jos tietokoneeseesi on asennettu Microsoft Office, voit tutkia käytettävissäsi olevia objekteja, ominaisuuksia ja metodeja Visual Basic Object Browserilla.

Seuraavissa jaksoissa opit kirjoittamaan Visual Basic -ohjelman, joka tarkistaa tekstiruudussa olevan tekstin oikeinkirjoituksen Word-objektin avulla. Voit huomata, että palvelinsovelluksen luovuttamat objektit, ominaisuudet ja metodit vastaavat tavallisesti sovelluksen valikkokomentoja ja valintaikkunoiden vaihtoehtoja.

Visual Basic Object Browser

Object Browserilla tutkitaan järjestelmän objekteja.

Visual Basic Object Browser on selausohjelma, jolla on kaksi käyttöä: se näyttää objektit, ominaisuudet ja metodit, jotka liittyvät Visual Basic -ohjelmointiympäristössä käsittelemääsi ohjelmaan tai jotka ovat käytettävissäsi järjestelmään asennettujen palvelinsovellusten ansiosta. Seuraavassa tehtävässä tarkastellaan Object Browserilla Wordin luovuttamia Automation-objekteja.

HUOMAA Kuvat esittävät Microsoft Word 97:ään sisältyvää Word 8.0 -objektikirjastoa. Ellei sinulla ole tätä Wordia, tutki Object Browserilla järjestelmäsi muita sovellusobjekteja.

Word-objektien katseleminen Object Browserilla

1. Valitse File-valikosta komento New Project ja napsauta OK.

   MyMusicEIS-sovellus katoaa näytöstä, ja uusi lomake tulee näkyviin. Sinun pitää lisätä siihen viittaus Word-objektikirjastoon.

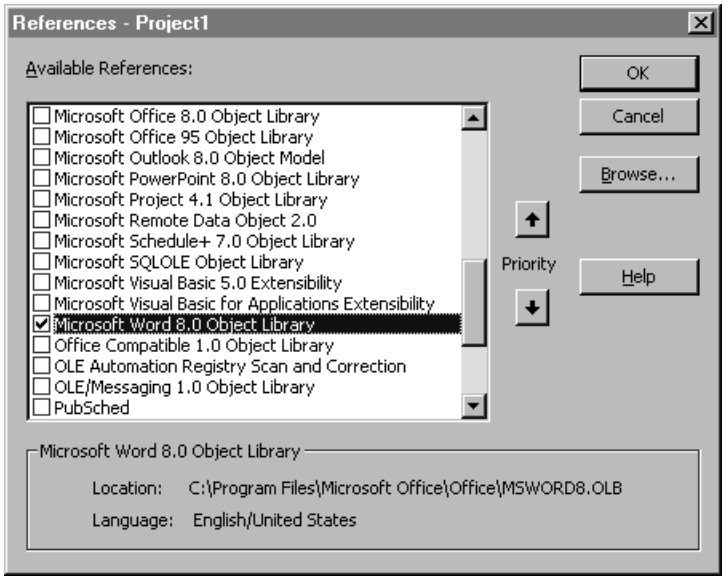
2. Valitse Project-valikosta komento References.

   References-valintaikkunassa voit lisätä projektiin viittauksia objektikirjastoihin.

   References-valintaikkuna avautuu, jotta voit sisällyttää projektiin viittauksia, jotka kohdistuvat käytettävissä oleviin objektikirjastoihin. Viittausten lisääminen projektiin ei yhtään suurenna käännettyä ohjelmatiedostoa, mutta mitä enemmän viittauksia on, sitä kauemmin Visual Basicilta menee ohjelman kääntämiseen. Siksi Visual Basic lisää viittauksia Automation-objektikirjastoihin vain pyydettäessä.

3. Valitse valintaruutu, joka on vaihtoehdon Microsoft Word 8.0 Object Library edellä.

   Sinun pitää ehkä vierittää luetteloa löytääksesi kirjaston (viittaukset ovat aakkosjärjestyksessä). Näytön pitäisi olla seuraavan sivun kuvan mukainen.

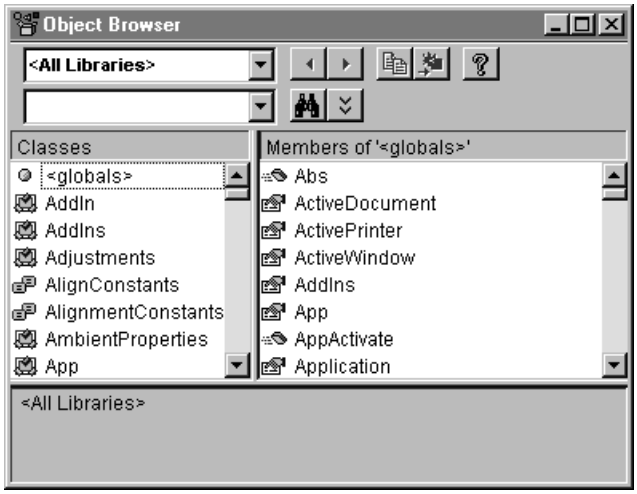


4 Sulje valintaikkuna ja lisää viittaus projektiin napsauttamalla OK.

Nyt olet valmis kokeilemaan Object Browseria.

5 Valitse View-valikosta komento Object Browser.

Object Browserin ikkuna avautuu seuraavan kuvan näköisenä:



Ikkunassa on avattava Project/Library-luetteloruutu, jolla voit tuoda esiin projektiin kuuluvat objektikirjastot. Siinä on myös avattava Search Text -luetteloruutu avainsanahakuja varten sekä Classes-luetteloruutu, josta voit

valita tutkittavan objektin. Kun valitset Classes-luetteloruudusta objektin, sen metodit, ominaisuudet ja tapahtumat näkyvät Members-luetteloruudussa.

6 Avaa Project/Library-luetteloruutu napsauttamalla sen alanuolipainiketta ja valitse Word-objektikirjasto.

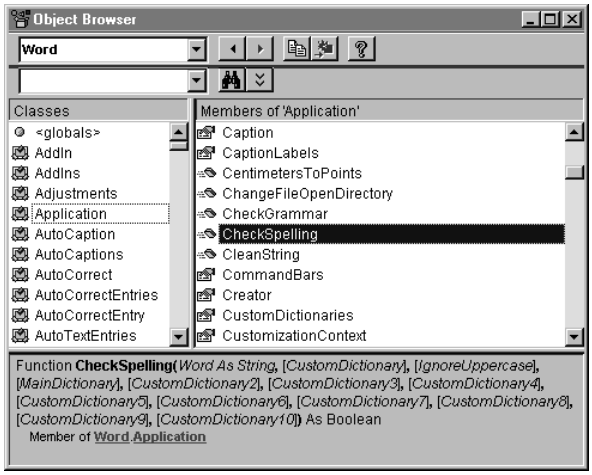
Wordin luovuttamien Automation-objektien nimet ilmestyvät Classes-luetteloruutuun.

7 Vieritä Classes-luetteloa alaspäin ja napsauta Application- eli sovellus-objektia.

Näet Application-objektiin liittyvien metodien ja ominaisuuksien nimet Members-luetteloruudussa. Nämä ovat komentoja, jotka Word tarjoaa asiakirjoissa olevien tietojen käsittelemistä varten.

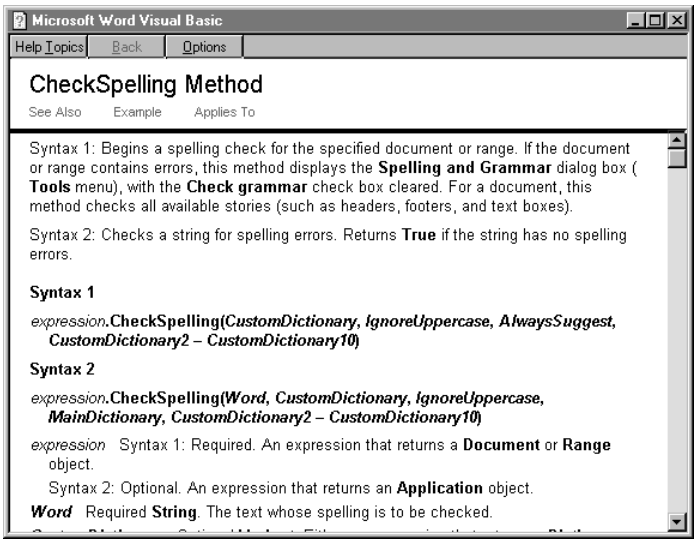
8 Valitse Members-luetteloruudusta metodi CheckSpelling.

CheckSpelling-metodin kutsun muoto näkyy Object Browserin ikkunan alareunassa. Tämä metodi käynnistää Wordin oikeinkirjoituksen tarkistimen, toiminnon, jota voidaan kutsua Visual Basicista. Metodin kieliopin kuvauksesta ilmenevät asetukset, jotka vaikuttavat tarkistuksen suorittamiseen. Object Browserin ikkunan pitäisi olla tällainen:



9 Napsauta Object Browserin ikkunan yläosassa olevaa kysymysmerkki-painiketta.

Visual Basic näyttää CheckSpelling-metodin ohjetiedoston seuraavalla sivulla olevan kuvan kaltaisena.



Objektikirjaston ohjejärjestelmä kuvaa palvelinsovelluksen luovuttamia ominaisuuksia ja metodeja.

Ohjetiedostossa on tarkkoja tietoja siitä, miten Word-objektikirjaston ominaisuuksia ja metodeja käytetään. Monet palvelinsovellukset sisällyttävät objektikirjastoihinsa nämä tiedot, jotta Automation-tekniikkaa käyttävät ohjelmoijat voivat hyödyntää sovelluksen ohjelmoitavia toimintoja. Objektikirjaston ohjejärjestelmä on erinomainen tapa oppia lisää Automation-ohjelmoinnista.

10 Lue CheckSpelling-metodia koskeva aihe ja sulje ohjetiedosto.

11 Napsauta Object Browserin otsikkorivin sulkemispainiketta.

Näin olet saanut jonkinlaisen käsityksen Automation-objekteista. Nyt on aika pistää Wordin CheckSpelling-metodi töihin

Wordin CheckSpelling-metodin käyttö

Kun haluat käyttää Wordin CheckSpelling-metodia Visual Basic -ohjelmassa, sinun pitää suorittaa seuraavat vaiheet. Koska toimintatapa on useimpien sovellusobjektien yhteydessä samanlainen, voit soveltaa näitä ohjeita melkein mihin tahansa ohjelmaan, joka tukee Automation-tekniikkaa.

VIHJE Wordin Visual Basic -kieli on niin hyödyllinen ja tehokas, että kirjoitin sen piirteitä ja kykyjä kuvaavan *Step by Step* -kirjan Chris Kinatan, arvostetun Word-tuntijan, kanssa. Jos olet kiinnostunut oppimaan lisää tästä aiheesta, tutustu Michael Halvorsonin ja Chris Kinatan kirjaan *Microsoft Word/Visual Basic Step by Step* (Microsoft Press, 1997).

Vaihe 1 Lisää projektiisi viittaukset välttämättömiin objektikirjastoihin References-komennolla.

Vaihe 2 Kirjoita Visual Basic -ohjelma. Luo Dim-lauseella objektimuuttuja tapahtuma-aliohjelmaan, jossa aiot käyttää Automation-objektia, ja lataa Automation-objekti muuttujaan CreateObject-funktiolla:

```
Dim X As Object                    'objektimuuttujan nimi on X
Set X = CreateObject("Word.Application")
```

Vaihe 3 Käytä tapahtuma-aliohjelmassa Automation-objektin metodeja ja ominaisuuksia. Katso Object Browserin ohjeista tai palvelinsovelluksen oppaista, miten metodeja ja ominaisuuksia pitää kutsua.

```
X.Visible = False                  'piilota Word
X.Documents.Add                    'avaa uusi asiakirja
X.Selection.Text = Text1.Text      'kopioi tekstiruutu asiakirjaan
X.ActiveDocument.CheckSpelling     'aloita oikoluku
Text1.Text = X.Selection.Text      'kopioi tulokset takaisin
```

Vaihe 4 Kun et enää tarvitse palvelinsovellusta, lopeta se ja vapauta objektimuuttuja muistin säästämiseksi:

```
X.Quit                              'lopeta Word
Set X = Nothing                    'vapauta objektimuuttuja
```

Seuraava tehtäväsi on kirjoittaa sovellus, joka tarkistaa Visual Basicin tekstiruudussa olevan tekstin oikeinkirjoituksen Wordilla. Ohjelma muodostetaan Visual Basicilla, mutta se käyttää Wordin toimintoja Automation-tekniikalla.

HUOMAA Tässä oletetaan, että järjestelmääsi on asennettu Word 97 tai Office 97 (johon Word 97 sisältyy).

Oikeinkirjoituksen tarkistimen luominen

1. Valitse Project-valikosta komento References. Tarkista, että vaihtoehdon Microsoft Word 8.0 Object Library valintaruudussa on rasti, ja napsauta OK.

   Microsoft Word 8.0 -objektikirjaston kautta pääset käyttämään Word-sovelluksen luovuttamia objekteja, metodeja ja ominaisuuksia. Lisäsit viittauksen tähän objektikirjastoon aiemmin tässä harjoituksessa.

HUOMAA Objektikirjastoihin kohdistuvat viittaukset on lisättävä jokaiseen uuteen projektiin References-komennolla.

Luo lomake oikeinkirjoituksen tarkistinta varten ja lisää tarvittavat ohjelmalauseet.

2 Muuta lomake tavanomaisen valintaikkunan kokoiseksi.

Teksti-työkalu

3 Napsauta Teksti-työkalua ja lisää lomakkeen yläosaan pitkä tekstiobjekti.

Tähän objektiin kirjoitetaan ohjelman käyttöohje.

Tekstiruutu-työkalu

4 Napsauta Tekstiruutu-työkalua ja lisää lomakkeen keskelle leveä nelirivinen tekstiruutuobjekti.

Tähän objektiin kirjoitetaan teksti, jonka oikeinkirjoitus tarkistetaan.

Komentopainike-työkalu

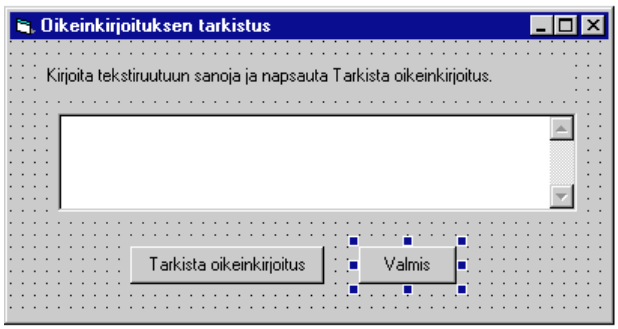
5 Napsauta Komentopainike-työkalua ja lisää tekstiruutuobjektin alapuolelle kaksi komentopainiketta.

Ensimmäisellä painikkeella käynnistetään Word ja kutsutaan Check-Spelling-metodia, joka tarkistaa tekstiruudussa olevan tekstin oikeinkirjoituksen. Toisella painikkeella lopetetaan ohjelma.

6 Määritä ohjelman objektien ominaisuudet näin:

Objekti	Ominaisuus	Asetus
Form1	Caption	”Oikeinkirjoituksen tarkistus”
Label1	Caption	”Kirjoita tekstiruutuun sanoja ja napsauta Tarkista oikeinkirjoitus.”
Text1	MultiLine ScrollBars Text	True 2 - Vertical (Tyhjä)
Command1	Caption	”Tarkista oikeinkirjoitus”
Command2	Caption	”Valmis”

Kun olet määrittänyt ominaisuudet, lomakkeen pitäisi näyttää tällaiselta:



7 Kaksoisnapsauta Check Spelling -painiketta, jotta näet tapahtuma-aliohjelman Command1_Click. Kirjoita siihen seuraavat lauseet:

```
Dim X As Object                    'luo   Word-objektimuuttuja
Set X = CreateObject("Word.Application")
X.Visible = False                  'piilota Word
X.Documents.Add                    'avaa  uusi  asiakirja
X.Selection.Text = Text1.Text      'kopioi  teksti  asiakirjaan
X.ActiveDocument.CheckSpelling     'tarkista   oikeinkirjoitus
Text1.Text = X.Selection.Text      'kopioi  tulokset  takaisin
X.ActiveDocument.Close SaveChanges:=wdDoNotSaveChanges
X.Quit                              'lopeta Word

Set X = Nothing                    'vapauta   objektimuuttuja
```

Nämä lauseet luovat tapahtuma-aliohjelmassa Word Automation -objektin, määrittävät muutamia sen ominaisuuksia, kutsuvat sen metodeja ja vapauttavat sille varatun muistin. Word käynnistyy automaattisesti, kun objektimuuttujaan viitataan ensimmäisen kerran. Tekstiruudun sisältö kopioidaan Word-asiakirjaan sijoittamalla se objektin Selection.Text-ominaisuuteen.

Kun CheckSpelling-metodi suoritetaan, Word käynnistää oikeinkirjoituksen tarkistimen ja tutkii, onko tekstissä kirjoitusvirheitä. Jos virhe löytyy, Wordin Spelling-valintaikkuna avautuu, ja käyttäjä voi korjata virheen. Word tutkii tekstiruudun jokaisen sanan, vaikka tekstiä olisi useita rivejä. Tarkistuksen päätyttyä ohjelma kopioi korjatut sanat takaisin Visual Basicin tekstiruutuun ja lopettaa Wordin. Tapahtuma-aliohjelman lopussa oleva Set-lause vapauttaa objektimuuttujalle varatun muistitilan.

8 Sulje Command1_Click-aliohjelma ja kaksoisnapsauta lomakkeen Done-painiketta.

9 Kirjoita tapahtuma-aliohjelmaan **End** ja sulje Code-ikkuna.

Tekstiruudun oikeinkirjoituksen tarkistava ohjelma on nyt valmis.

10 Valitse komento Save Form1 As ja tallenna lomake kansioon \Vb5Sbs\Less12 nimellä **MyUseWord.frm**. Valitse komento Save Project As ja tallenna projekti nimellä **MyUseWord.vbp**.

Käynnistä ohjelma, jotta näet, miten Automation-tekniikka toimii.

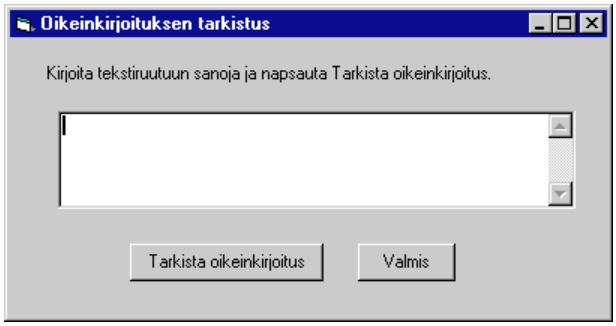
MyUseWord-ohjelman suorittaminen

Käynnistä-painike

1 Napsauta työkalurivin Käynnistä-painiketta.

Ohjelma käynnistyy ohjelmointiympäristön alaisuudessa, kuten seuraavan sivun kuvasta näkyy.

Valmis UseWord-ohjelma on kansiossa \Vb5Sbs\Less12.

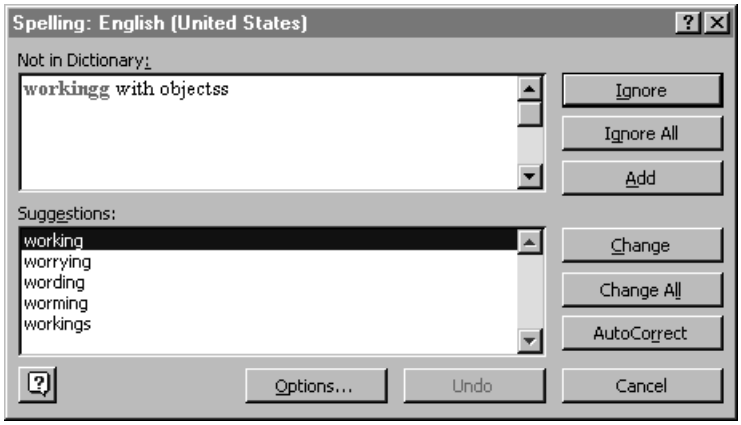


2. Kirjoita tekstiruutuun **workingg with objectss**.

   Kirjoita sanat *working* ja *objects* tahallisesti väärin.

3. Napsauta Check Spelling -painiketta.

   Visual Basic luo Automation-objektin ja käynnistää Wordin. Hetken kuluttua Wordin Spelling-valintaikkuna avautuu ja näyttää ensimmäisen sanan, jota ei ole sanastossa. Valintaikkunan pitäisi olla tällainen:



4. Korjaa ensimmäinen virhe napsauttamalla Change. Kun Word korostaa toisen kirjoitusvirheen, korjaa sekin napsauttamalla Change.

   Spelling-valintaikkuna sulkeutuu, ja tekstiruudussa näkyvät sanat muuttuvat hetken kuluttua oikeiksi.

5. Lopeta ohjelma napsauttamalla Valmis.

Onnittelut! Olet juuri oppinut kutsumaan Automation-tekniikalla Wordin moduulia, joka auttaa kirjoitusvirheiden korjaamisessa. Tämä on vasta alkua sille, mitä voit tehdä Visual Basicilla ja sovellusobjekteilla.

Lisätietoja: Etsintä ja korvaus käyttämällä Wordia

Jos tietokoneeseesi on asennettu Word 97, kokeile toista Automation-tekniikkaa, jossa käytetään Wordia tekstin etsimiseen ja korvaamiseen. Seuraava ohjelma etsii ja korvaa tekstiruudussa olevaa tekstiä Wordin Find-objektilla. Voit helposti muuttaa aliohjelmaa muihin tarkoituksiin, esimerkiksi tekstitiedoston tai tietokannan muokkaamiseen.

WdSearch-ohjelman suorittaminen

Lataa WdSearch-ohjelma levyltä ja käynnistä se.

Avaa projekti -painike

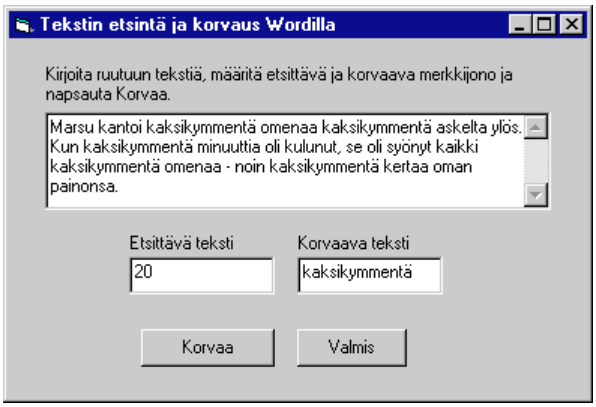
Käynnistä-painike

1. Napsauta työkalurivin Avaa projekti -painiketta.
2. Valitse kansio \Vb5Sbs\Less12 ja kaksoisnapsauta tiedostoa WdSearch.
3. Napsauta Työkalurivin Käynnistä-painiketta.

   WdSearch käynnistyy ohjelmointiympäristön alaisuudessa.
4. Kirjoita tekstiruutuun seuraava teksti:

   Marsu kantoi 20 omenaa 20 askelta ylös. Kun 20 minuuttia oli kulunut, se oli syönyt kaikki 20 omenaa - noin 20 kertaa oman painonsa.
5. Paina SARKAINNÄPPÄINTÄ ja kirjoita Etsittävä teksti -ruutuun 20.
6. Paina SARKAINNÄPPÄINTÄ ja kirjoita Korvaava teksti -ruutuun **kaksikymmentä**.
7. Käynnistä Microsoft Wordin etsintä- ja korvaustoiminto napsauttamalla Korvaa.

   Etsinnän ja korvauksen tulokset näkyvät kohta tekstiruudussa.



8. Lopeta ohjelma napsauttamalla Valmis.

Automation-lauseiden tutkiminen

Tarkastele Automation-tekniikkaa käyttäviä lauseita, jotka käynnistävät Wordin etsintä- ja korvaustoiminnon.

1. Kaksoisnapsauta WdSearch-lomakkeen Korvaa-painiketta.

   Tapahtuma-aliohjelma Command1_Click ilmestyy Code-ikkunaan. Sen Automation-lauseet ovat samankaltaisia kuin edellä UseWord-ohjelmassa. CheckSpelling-metodi on vaihdettu lauseeseen, joka aloittaa etsimisen ja korvaamisen.

2. Vieritä näkyviin aliohjelman keskivaiheilla oleva With-rakenne. Siinä on tällaiset lauseet:

```
With X.Selection.Find      'käynnistä  etsintä  ja  korvaus
    .ClearFormatting       'älä  välitä  muotoilusta
    .Text = Text2.Text     'etsittävä   teksti
    .Replacement.ClearFormatting
    .Replacement.Text = Text3.Text 'korvaava   teksti
    .Execute Replace:=wdReplaceAll, Forward:=True, _
        Wrap:=wdFindContinue
End With
```

   With-rakenne on tiivis tapa käsitellä ohjelmassa useita objekteja samanaikaisesti. Tämä With-rakenne viittaa X.Selection.Find-objektiin viidesti. Neljä ensimmäistä lausetta määrittävät ominaisuuksia, jotka valmistelevat Wordin etsintää ja korvausta varten. ClearFormatting-ominaisuus tarkoittaa, että Word ei kiinnitä huomiota etsittävän tekstin muotoiluun — tämä vastaa Wordin Find-valintaikkunan No Formatting -asetusta.

   Seuraavaksi Text-ominaisuudella määritetään etsittäväksi tekstiksi Search-tekstiruudun sisältö (Text2.Text). Replacement.ClearFormatting-ominaisuus merkitsee, että korvaavan tekstin muotoilulla ei ole väliä, ja Replacement.Text-ominaisuus määrittää korvaavan tekstin (Text3.Text). Lopuksi Execute-metodi aloittaa etsinnän ja korvauksen koko asiakirjassa.

3. Sulje Code-ikkuna. Olet nyt suorittanut tämän harjoituksen kaikki osat.

Jos haluat harjoitella ohjelmointitaitojasi

Lue asiakirja Review & Practice 4, joka on tallennettu kansion \Vb5Sbs\Review tiedostoon Rev4.doc. Se sisältää vaiheittain eteneviä tehtäviä, joilla voit testata ohjelmointitaitosi ja kerrata lukujen 9–12 sisällön. Tehtävissä muodostetaan tietokannan katseluohjelma, joka perustuu Wordin Automation-tekniikkaan.

Jos haluat lopettaa Visual Basicin

➤ Valitse File-valikosta komento Exit.

Jos näyttöön tulee Save-valintaikkuna, napsauta Yes.

Harjoituksen tiivistelmä

Tehtävä	Ohje	Painike
Lisätä ohjelman sovellus- eli OLE-objekti	Varaa OLE-objektille tilaa lomakkeesta OLE-työkalulla ja kaksoisnapsauta objektia Insert Object -valintaikkunassa.	
Luoda linkitetty sovellusobjekti	Valitse Insert Object -valintaikkunasta Link-valintaruutu.	
Luoda upotettu sovellusobjekti	Älä valitse Insert Object -valintaikkunasta Link-valintaruutua. (Insert Object -valintaikkunan avulla lisätyt objektit ovat oletuksen mukaan upotettuja eli sijaitsevat Visual Basic -ohjelmassa.)	
Lisätä tiedoston osa sovellusobjektiin	Valitse tarvitsemasi tiedot sovelluksella, siirrä osoitin sovellusobjektin päälle Visual Basicissa, napsauta hiiren oikeaa painiketta ja valitse komento Paste Special.	
Päivittää linkit OLE1-nimiseen objektiin ohjelman käynnistyttyä	Kirjoita Form_Load-aliohjelmaan lause `OLE1.Update`	
Valita sovelluskirjasto	Valitse Project-valikosta komento References. Valitse tarvitsemasi sovelluksen nimen edellä oleva valintaruutu.	
Tutkia sovellusobjekteja, joissa on Automation-tuki	Valitse View-valikosta komento Object Browser. Valitse tutkittava objekti Project/Library-luettelosta.	

Lisätä ohjelmaan objektimuuttuja	Käytä Dim- ja Set-lauseita. Esimerkki:
	`Dim X As Object`
	`Set X = CreateObject("Word.Application")`	

Käyttää sovelluksen toimintoja Automation-tekniikalla	Luo objektimuuttuja ja viittaa sen metodeihin tai ominaisuuksiin. Esimerkki:
	`X.CheckSpelling`
	`X.Quit`	

Vapauttaa objektimuuttujalle varattu muisti	Käytä Set-lausetta, jossa on muuttujan nimi ja avainsana Nothing. Esimerkki: `Set X = Nothing`	

Kun tarvitset tietoja aiheesta	Valitse Visual Basicin Help-valikosta Microsoft Visual Basic Help Topics, valitse Index-välilehti ja...
Sovellusobjekti	Kirjoita **OLE Container control**.
Object Browser	Kirjoita **Object Browser**.
Sovellusobjektien ohjelmointi	Kirjoita **OLE Automation**.

Mitä seuraavaksi?

Nyt olet suorittanut koko Visual Basic -ohjelmointikurssin. Sinun kannattaa seuraavaksi etsiä Visual Basic 5:n ohjejärjestelmästä kiinnostavia aiheita ja tutustua Visual Basic 5:n mukana toimitettaviin esimerkkiohjelmiin. Niissä käsitellään monia kiinnostavia ja hyödyllisiä asioita, esimerkiksi Office 97:n Visual Basic -makroja, tehokkaita Windows API -funktioita, Internet-tuen lisäämistä, ohjelmien koon ja nopeuden optimoimista sekä asennusohjelman luomista. Nyt kun hallitset perusteet, voit huoletta edetä vaativampiin tehtäviin.

Seuraavista kirjoista on apua, kun perehdyt Visual Basiciin tarkemmin:

- Jos haluat tietää, miten Visual Basic for Applicationsilla hallitaan asiakirjoja ja käytetään Internetiä, hanki Michael Halvorsonin ja Chris Kinatan teos *Microsoft Word 97/Visual Basic Step by Step* (Microsoft Press, 1997), ISBN 1-57231-388-9.
- Jos opiskelet mieluiten kokeilemalla todellisia sovelluksia ja tutkimalla esimerkkiohjelmia, suosittelen John Clark Craigin ja Jeff Webbin teosta *Microsoft Visual Basic 5.0 Developer's Workshop, Fourth Edition* (Microsoft Press, 1997), ISBN 1-57231-436-2.
- Jos haluat perehtyä keskitason ja vaativiin Visual Basicia koskeviin aiheisiin multimediaohjelmointikurssilla, johon sisältyy videoita, ääntä, animaatioita ja lähdekoodia, hanki *Mastering Visual Basic 5.0* (Microsoft Corporation, 1997).
- Tarkkoja tietoja ammattilaisten ohjelmointimenetelmistä ja Windows API -funktioista on Bruce McKinneyn teoksessa *Tehokäyttäjän opas Visual Basic 4*.

VIHJE Jos haluat nähdä Microsoft Pressin kirjojen täydellisen luettelon, tutustu Microsoft Pressin Web-sivuihin. Niiden osoite on http://www.microsoft.com/mspress/ ja http://www.satku.fi.

Ajatteleminen ohjelmoijan tavalla

Jos ohjelmoiminen on sinulle aivan uutta, ihmettelet ehkä, mitä ohjelmat ovat ja miten Visual Basicilla tehdään ohjelmia. Tässä liitteessä kuvataan ohjelman kehitysprosessin yleisiä vaiheita. Voit lukea sen ennen kuin teet ensimmäisen ohjelmasi harjoituksessa 1 tai myöhemmin, kun haluat tietää lisää ohjelmoinnin käsitteistä.

Mikä on ohjelma?

Visual Basic –ohjelma on Windows–sovellus.

Ohjelma on joukko käskyjä, jotka yhdessä saavat tietokoneen suorittamaan tehtävän, esimerkiksi muokkaamaan elektronista kuvaa tai käsittelemään verkon tiedostoja. Ohjelma voi olla pieni apuohjelma, joka esimerkiksi laskee asuntolainan kuoletukset, tai suuri sovellus, kuten Microsoft Excel. *Visual Basic -ohjelma* on sovellus, joka on kirjoitettu Microsoft Windows -ympäristöä varten Visual Basic -ohjelmointijärjestelmällä. Kun aiot kirjoittaa Visual Basic -ohjelman, sinun tulee selvittää, mitä sen pitää tehdä ja millaiselta sen näytön eli *käyttöliittymän* pitää näyttää, ja sitten rakentaa ohjelma Visual Basic -ympäristössä. Tutkitaan näitä vaiheita vähän tarkemmin.

Ohjelman suunnitteleminen

Ohjelmoinnin ensimmäinen vaihe on suunnittelu.

Ohjelmoinnin ensimmäinen vaihe on selvittää tarkasti, mitä ohjelman on tehtävä. Tämä kuulostaa yksinkertaiselta tai päivänselvältä, mutta ilman selkeää tavoitetta parhaankin ohjelmoijan on vaikea saada aikaan tyydyttävä sovellus. Ohjelman suunnitteleminen on vähän kuin kutsujen järjestämistä: sinun pitää valmistautua. Sinun tarvitsee valmistella ruokalista, kutsua ystäväsi, ostaa ruoka-aineet ja (todennäköisesti) siivota asuntosi. Vaikka vieraat voivat viihtyä kutsuilla, vaikka

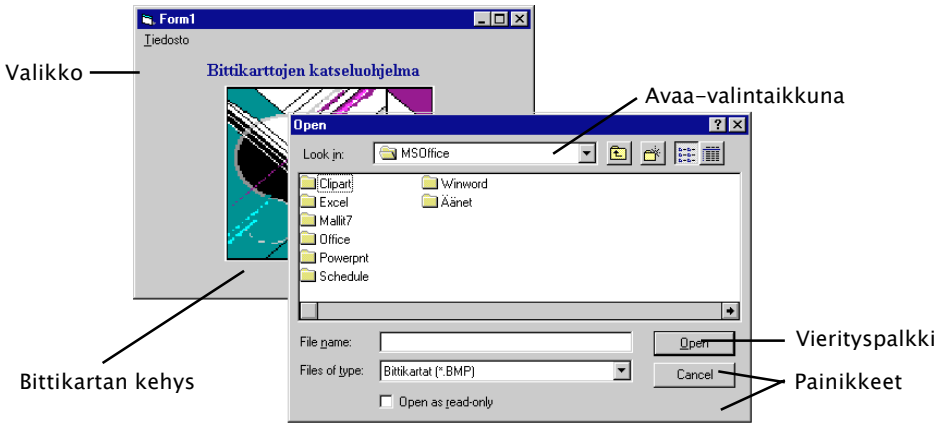
he vain saapuvat varoittamatta ja tuovat tarvikkeet tullessaan, tietokoneohjelmasta ei tavallisesti tule kovin ihmeellinen, jos se vain kasataan liikoja suunnittelematta.

Kun opettelet ohjelmoimaan, sinun kannattaa varata aikaa ongelman ratkaisun pohtimiseen ennen kuin istut tietokoneesi ääreen. Huolellinen suunnittelu lyhentää ohjelman kehittelemiseen tarvittavaa aikaa ja todennäköisesti parantaa lopputulosta. Suunnittelun yksi osa saattaa olla se, että laadit *algoritmiksi* kutsutun luettelon. Se kuvaa järjestyksessä osatehtäviä, jotka ohjelman pitää suorittaa.

Käyttöliittymän suunnitteleminen

Käyttöliittymä sisältää ohjelman kaikki ikkunat ja niiden osat.

Kun ohjelmallasi on selvä tavoite, sinun kannattaa miettiä tarkoin, millainen ohjelmasta pitää tulla ja miten sen pitää käsitellä tietoja. Ohjelmassa käytettävien ikkunoiden ja valintaikkunoiden kokonaisuutta sanotaan *käyttöliittymäksi*. Se sisältää kaikki valikot, valintaikkunat, painikkeet, objekti ja kuvat, jotka käyttäjät näkevät. Visual Basicilla pystyt luomaan kaikki Windows-sovelluksessa tarvittavat osat nopeasti ja tehokkaasti. Useita näistä osista näkyy seuraavassa Windows 95 -ohjelmassa, jonka kirjoitin Visual Basicilla:



Jos olet käyttänyt Windows-sovelluksia, useimmat käyttöliittymän osat ovat sinulle tuttuja. *Valikoissa* on sovelluksen *komentoja*; *valintaikkunat*, *painikkeet* ja *osoitin* ovat keinoja antaa ohjelmalle komentoja ja tietoja ja *ikkunoiden* ja *vierityspalkkien* avulla käyttäjä voi katsella ja selata näytössä olevia tietoja. Jotta Visual Basic -ohjelmastasi tulisi mahdollisimman helppokäyttöinen, on tärkeää, että käytät Windows-käyttöliittymän osia totutulla ja ennustettavalla tavalla. Microsoft on julkaissut joukon Windows-sovellusten suunnitteluohjeita, ja tässä kirjassa kuvataan tärkeimpiä niistä. Koska useimmat Windows-sovellukset on tarkoitettu muille käyttäjille kuin itse ohjelmoijalle, on tärkeää, että teet ohjelmasi käyttöliittymästä samankaltaisen kuin muissa sovelluksissa.

Kätevän käyttöliittymän suunnittelemista kuvataan harjoituksissa 1 ja 2.

HUOMAA Jos haluat tutustua Windows 95 -sovellusten virallisiin käyttöliittymän suunnitteluohjeisiin, katso teosta *The Windows Interface Guidelines for Software Design* (Microsoft Press, 1995).

Hyödyllisiä kysymyksiä

Kun suunnittelet Visual Basic -ohjelmaa, sinun kannattaa ehkä hahmotella käyttöliittymä, jonka aiot esittää käyttäjille. Tämä auttaa käyttöliittymän sekä ohjelman toimintojen suunnitelemisessa. Suosittelen, että esität itsellesi seuraavanlaisia kysymyksiä:

Näistä kysymyksistä on apua suunnitellessasi ohjelmaa.

- Mikä on ohjelman tehtävä tai käyttötarkoitus?
- Kuka ohjelmaa käyttää?
- Miltä ohjelman pitää näyttää käynnistymisen jälkeen?
- Millaisia tietoja käyttäjän tulee antaa ohjelmalle?
- Miten ohjelma käsittelee syöttötiedot?
- Millaisia tuloksia ohjelma tuottaa?

Jo ennen kuin tiedät, *miten* ohjelma suorittaa tehtävän, sinulla pitäisi olla selvä käsitys siitä, *mitä* ohjelman pitää tehdä ja *miltä* sen pitää näyttää. Kun olet huolehtinut näistä valmisteluista, voit alkaa kehitellä ohjelmaa Visual Basicilla.

Ohjelman luominen Visual Basicilla

Windows-ohjelman luomiseen Visual Basicilla liittyy kolme vaihetta:

1. Muodosta käyttöliittymä Visual Basicin työkaluilla.
2. Määritä käyttöliittymän osien *ominaisuudet* sopiviksi.
3. Kirjoita ohjelmalauseet, jotka liittyvät käyttöliittymän osiin.

Windows-ohjelman luominen Visual Basicilla on hämmästyttävän helppoa, kuten huomaat harjoituksessa 1. Jos olet käyttänyt Paintin kaltaista piirrosohjelmaa, hallitset jo monia käyttöliittymän laatimisessa tarvittavia taitoja. Sinun tarvitsee vain valita työkalu Visual Basicin työkaluruudusta ja ”piirtää” käyttöliittymän osa napsauttamalla ja vetämällä hiirtä. Napsauta kohtaa, johon haluat sijoittaa osan yhden kulman ja määritä sopivankokoinen suorakulmio vetämällä hiirtä. Kun olet luonut käyttöliittymän osan, kuten tekstiruudun, määritä sen ominaisuudet. Voit esimerkiksi muuttaa tekstiruudun tekstin lihavoiduksi, kursivoiduksi tai alleviivatuksi.

Liitä lopuksi käyttöliittymän yhteen tai useaan osaan ohjelmalauseet kirjoittamalla ne Visual Basicin Code-ikkunaan. Ohjelmalauseilla pystyt ohjaamaan ohjelman toimintaa paremmin kuin käyttämällä vain valmiiksi määritettyjä objekteja. Ohjelmalauseet määräävät tarkasti, miten ohjelma lukee, käsittelee ja tulostaa tietoja.

Visual Basic -kieli

Visual Basic -ohjelmointikieli sisältää useita satoja lauseita, funktioita ja erikoismerkkejä, mutta selviydyt useimmista ohjelmointitehtävistä muutamalla kymmenellä helposti muistettavalla avainsanalla. Visual Basic -kieltä käytetään muuallakin kuin Visual Basic 5 -ohjelmointiympäristössä: Microsoft Visual Basic for Applications sisältyy Microsoft Exceliin, Wordiin, Accessiin, PowerPointiin, Projectiin ja moniin muihin Windows-sovelluksiin. Jos olet käyttänyt näiden sovellusten uusimpia versioita tai olet tutustunut Basic-kieleen aikaisemmin, tunnet jo monia Visual Basic 5:n avainsanoja, lauseita ja funktioita.

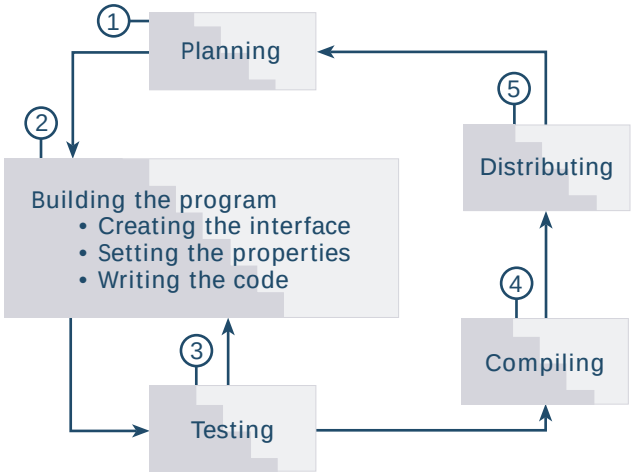
HUOMAA Visual Basic -ohjelmointikielen rakennetta ja sisältöä koskevia sääntöjä sanotaan *kieliopiksi* tai *syntaksiksi*. Sana *syntaksi* tulee kreikan kielen verbistä *syntattein,* joka tarkoittaa armeijan johtamista. Kun kirjoitat Visual Basic -ohjelman, annat kääntäjälle komentoja avainsanojen muodossa.

Basicin alkuperä

Alkuperäisen Basic-kielen kehittivät John G. Kemeny ja Thomas E. Kurtz Dartmouth Collegessa vuonna 1963. Siitä tuli nopeasti suosittu opetuskieli yliopistoissa ja kouluissa, ja sen sovitti mikrotietokoneisiin Microsoftin perustaja ja johtaja Bill Gates 1970-luvun puolivälissä. Noiden aikojen jälkeen Basicista on tehty monia päivitettyjä versioita, esimerkiksi Microsoft QuickBasic ja MS-DOS QBasic. Vaikka Visual Basicin täysin graafinen kehitysympäristö ja ohjelmointikieli ovat kaukana varhaisista Basic-tulkeista, suuri osa alkuperäisen Basicin tyylikkyydestä ja yksinkertaisuudesta on edelleen tallella. Visual Basicin helppokäyttöisyys ja tehokkuus ovat tärkeimpiä syitä, joiden takia se valittiin Windows-sovellusten, esimerkiksi Excelin, ohjelmointikieleksi.

Ohjelman testaaminen, kääntäminen ja jakeleminen

Kun olet saanut ohjelmasta toimivan version, sinun pitää testata se huolellisesti, kääntää se suoritettavaan muotoon (itsenäiseksi Windows-ohjelmaksi) ja jakaa se käyttäjille. Jos päätät myöhemmin korjailla ohjelmaa, aloita koko prosessi alusta eli suunnittelemisesta ja uusien tavoitteiden asettamisesta. Näistä vaiheista muodostuu ohjelman elinkaari, joka on esitetty seuraavan sivun kuvassa.



Ohjelman testaaminen

Ohjelman testaaminen tarkoittaa sen moitteettoman toiminnan varmistamista erilaisissa käytännön tilanteissa. Ongelmaa, joka pysäyttää ohjelman tai estää sitä tuottamasta odotettuja tuloksia, sanotaan *virheeksi*. Jos olet käyttänyt jotakin ohjelmaa pitkän aikaa, olet varmaan törmännyt virheiden aiheuttamiin ongelmiin. Nyt kun *sinä* olet ohjelmoija, on sinun tehtäväsi poistaa ongelmat *ennen* kuin ne tuottavat harmia käyttäjälle.

Visual Basicissa on onneksi erinomaisia välineitä virheiden etsimistä ja korjaamista varten. Niitä kuvataan harjoituksessa 5.

Ohjelman kääntäminen

Suoritettavien tiedostojen tarkenne on .exe.

Kun olet saanut ohjelman valmiiksi ja testannut sen, voit kääntää sen suoritettavaksi tiedostoksi (.exe), joka toimii Microsoft Windows 95:ssä tai Windows NT:ssä. Suoritettavan tiedoston muodostaminen Visual Basicissa ei vaadi muuta kuin Make-komennon valitsemisen File-valikosta (katso lukua 1).

Jos aiot käyttää ohjelmaa vain omassa tietokoneessasi, suoritettavan tiedoston tekeminen on ohjelmointiprojektin viimeinen vaihe. Kun haluat käynnistää ohjelman, kaksoisnapsauta sen .exe-tiedostoa Windows Resurssienhallinnassa (Explorer) tai lisää Windowsin työpöydälle ohjelman pikakuvake.

Ohjelman jakeleminen

Visual Basic -ohjelmat tarvitsevat toimiakseen dynaamisia linkkikirjastoja (DLL).

Jos haluat jakaa uuden ohjelmasi ystäviesi tai työtovereidesi kanssa tai jos aiot myydä sitä, sinun pitää tallentaa se ja eräitä välttämättömiä aputiedostoja levykkeelle (tai verkkoon), jolta käyttäjät tai asiakkaat voivat kopioida ne. Jokainen Visual Basic -ohjelma tarvitsee toimiakseen yhden tai useita dynaamisia linkkikirjastoja (DLL). joten sinun täytyy kopioida välttämättömät tiedostot kiintolevyltä jakelulevykkeelle.

Tarvittavat DLL-tiedostot vaihtelevat sen mukaan, mikä käyttöjärjestelmä ohjelman käyttäjillä on ja mitä Visual Basicin ominaisuuksia ohjelma hyödyntää. Visual Basicin ohjeissa on lueteltu tiedostot, jotka tarvitaan jakelulevykkeillä, ja niissä selitetään myös, miten voit tehdä oman asennusohjelman Application Setup Wizardilla.

<> (erisuuruusoperaattori), 138
/ (jakolaskuoperaattori), *123,* 124–26, *130*
\~ (kokonaislukujen jakolasku-operaattori), *123,* 127–30, *130*
... (kolme pistettä), 77
“ (lainausmerkit), 111
– (negaatio-operaattori), *130*
< (pienempi kuin -operaattori), 138
<= (pienempi tai yhtä suuri kuin -operaattori), 138
, (pilkku), 164
; (puolipiste), 164
_ (rivinjatkomerkki), 141
= (sijoitusoperaattori), 109
() (sulkeet), 131
> (suurempi kuin -operaattori), 138
>= (suurempi tai yhtä suuri kuin -operaattori), 138
& (valintakirjain), 80, 81–82
– (vähennyslaskuoperaattori), *123,* 124–26
+ (yhteenlaskuoperaattori), *123,* 124–26, *130*
& (yhteenliittämisoperaattori), *123,* 127–30, 311
= (yhtäläisyysoperaattori), 138

A

Abs-funktio, 127
ActiveX-objektit
  asentaminen, 71–73
  luetteleminen valintaikkunassa, 72
  työkaluruudussa, 73
  Windows 95 -yhteensopivien luettelo, 74
Add Form -valintaikkuna, *188*
Add Form -komento, 191
Add Record -valintaikkuna, *321*
AddData-ohjelma, 321
AddItem-metodi, 59, 60, 148–49
AddNew-metodi, 319, 320
AdvOp-ohjelma, 127–29
ajastinobjektit, 176–81, 235–36, 237–38
ajonaikaiset virheet
  virheenkäsittelijät, 53, 203–9
  välttäminen, 53
  yleiskuvaus, 53, 151
algoritmit, luominen, 16, 356
Alignment-ominaisuus, 24
alilomakkeet, 197
aliohjelmat, 31. *Katso myös* tapahtuma-aliohjelmat; yleiskäyttöiset aliohjelmat
aloituslomakkeet, 214–18
And-operaattori, *143,* 144–45
animaatio
  ja koordinaatistot, 234–35
  kuvaus, 234
  lisääminen DragDrop-ohjelmaan, 236–39
  luominen, 236–40
  StepUp-ohjelmassa, 239
Appearance-ominaisuus, 64
argumentit
  arvo-, 271–72
  funktioissa, 115
  kuvaus, 114
  viittaus-, 264
asemaluetteloruudut, 48, 49, 50
asentaminen, ActiveX-objektit, 71–74
asiakassovellukset, 342
asiakirjakeskeiset sovellukset, 197
Askel-painike, 152
askellustila
  käyttö, 151–57
  valitseminen Keskeytä-painikkeella, 153
  valitseminen Stop-lauseella, 157
Atm-funktio, 127
Automation-tekniikka
  ohjelmoiminen sovellusobjekteilla, 341–50
  yleiskuvaus, 61, 341–42
AutoRedraw-ominaisuus, 288

avaaminen
Code-ikkuna, 28–29
Menu Editor, 78
Object Browser -ohjelma, 344
projektitiedostot, 6–7
avainsanat, kirjoittaminen, 29–30
avattavat luetteloruudut, 56

B

Backup-ohjelma, 326
Baseball-ohjelma, 294–97
Basic-kieli, historia, 359
.bas-tiedostot, 252
.bat-tiedostot, 302
Biblio.mdb-tiedosto
kuvaus, 312
näyttäminen Data Browserilla, 313–16
tietojen etsiminen, 316–19
tietueiden lisääminen, 319–22
tietueiden poistaminen, 322–24
varmistuskopion tekeminen, 325–26
bittikartan katseluikkuna, *357*
bittikarttakuvaobjektit, 64
.bmp-tiedostot, 48
Boolean-tietotyyppi, 118
BorderColor-ominaisuus, 214
BorderStyle-ominaisuus, 24, 214
BorderWidth-ominaisuus, 214
Browser-ohjelma
kirjoittaminen, 51
luominen, 49–50
objektien lisääminen, 49–50
ominaisuuksien muuttaminen, 50–51
suorittaminen, 52–53
Btrieve-tietokannat, 312
Buttons-ohjelma, 225
ByVal-avainsana, 271–72

C

Caption-ominaisuus, 11, 22–23, 25, 69, 178
Case-ohjelma, 150
Celsius-ohjelma, 176
CheckBox-objekti, 313
CheckSpelling-metodi, 345–46, 348–49
Circle-metodi, 218
Click-tapahtumat
ja MouseDown-tapahtumat, 219
ohjelmalauseiden katseleminen, 32–33
ohjelmalauseiden kirjoittaminen, 30–32
Close-avainsana, 303, 305, 306, 307
Cls-metodi, 289
Code-ikkuna
avaaminen, 28–29
kuvaus, 28, 29
ohjelmien katseleminen, 306–7
ohjelmien kirjoittaminen, 29–32
rivin pituus, 141
ComboBox-objekti, 313
Comdlg32.ocx-tiedosto, 72
CommandButton-objekti, 18, 19, 44, 45
CommonDialog-objekti, 90, 91, *91*
Components-valintaikkuna, 72–73
Connect-ominaisuus, 69
Constant-ohjelma, 122–23
Cos-funktio, 127
CtlArray-ohjelma, 169
Currency-tietotyyppi, 118

D

Data-ohjelma, 70, 119–21
Database-kokoelma, 282
DatabaseName-ominaisuus, 69, 315
DataField-ominaisuus, 70, 312, 315
dataobjektit
Biblio.mdb-tietokannassa, 315, 319, 322, 324, 326
kuvaus, 66
luominen, 67–68
ominaisuuksien muuttaminen, 69–70
tietokantaa muokattaessa, 71
tietokantaa näytettäessä, 68, 70

DataSource-ominaisuus, 69, 70, 312, 315
DataView-ohjelma
suorittaminen, 313–16
tallentaminen MyFindData-ohjelmaksi, 316–17
Date-funktio, 85, 88
Date$-funktio, 311
Date-tietotyyppi, 120
.dat-tiedostot, 302
Day-funktio, 88
dBase-tietokannat, 312
Debug-työkalurivi, 151–57, *152*
DelData-ohjelma, *324*
Delete-metodi, 322, 323
Description-ominaisuus, 208
Digitaalikello-ohjelma, 179–81
Dim-lause, 109, 110
DirListBox-objekti, *49*
Display As Icon -asetus, 63, *64*
DLL-tiedostot, 361
Do-silmukat
Celsius-ohjelmassa, 174–76
ehtolausekkeet, 173
loputtomat, 174
muoto, 173
Until-avainsana, 176
While-avainsana, 173, 176
yleiskuvaus, 173
DragDrop-aliohjelma
käyttö, 231–33
muokkaaminen, 237
yleiskuvaus, 227
DragDrop-ohjelma, 232, 236
DragIcon-ominaisuus, 227, 232
DragMode-ominaisuus, 227
DragOver-aliohjelma, 227
Drive-ominaisuus, 50
DriveErr-ohjelma, 205
DriveListBox-objekti, *49*
dynaamiset linkkikirjastot (DLL), 361
dynaamiset taulukot, 284, 290–93
DynArray-ohjelma, 293

E

ehtolausekkeet
Do-silmukoissa, 173
If…Then-rakenteissa, 138–40
järjestys, 140
kuvaus, 137
loogiset operaattorit, 143–45
operaattoriluettelo, 138
Select Case -rakenteissa, 146–47
vertailuoperaattorit, 138–40, 146–47
ehtorakenteet
If…Then, 138–45
kuvaus, 138
Select Case, 145–50
Else-avainsana, 139
Elseif-avainsana, 139
Enabled-ominaisuus, 341
End If -avainsana, 139
End Sub -lause, 29, 30
End-lause, 29–30
EndDoc-metodi, 198
EOF-avainsana, 303, 306
erisuuruusoperaattori (<>), 138
Err-objekti, 203, 206, 208
esittelyt, 109–11
et-merkki (&)
valintakirjaimissa, 80, 81–82
yhteenliittämisoperaattorina, *123,* 127–30, 311
Excel-kaaviot OLE-objekteina, 337–38
Excel-taulukot
käynnistäminen MyOleBid-ohjelmasta, 65–66
päivittäminen, 341
sovellusobjekteina, 64, 333, 336–37
.exe-tiedostot, 12, 35–37, 360
Exit For -lause, 172
Exit Sub -lause, 206
Exp-funktio, 127

F

FileCopy-lause, 325, 326
FileListBox-objekti, 49
FileName-ominaisuus, 50, 51, 302
FillColor-ominaisuus, 214
FillStyle-ominaisuus, 214
Filter-ominaisuus, 94, 95
FinalErr-ohjelma, 207
Find-objekti, 351
FixArray-ohjelma, 289
Flags-ominaisuus, 96–97
Font-ominaisuus, 24–25
FontBold-ominaisuus, 198, 221, 224
FontItalic-ominaisuus, 221
FontName-ominaisuus, 198
FontSize-ominaisuus, 164, 165–66, 198
Fontti-valintaikkuna, 24, *24,* 89
FontUnderline-ominaisuus, 221
For…Next-silmukka
- monimutkainen, 166–70
- muoto, 162
- MyFixArray-ohjelmassa, 287, 288–89
- ominaisuuksien muuttaminen, 164–66
- tiedostojen avaaminen, 167–69
- tietojen näyttäminen, 163–64
- yleiskuvaus, 161–62

For Each…Next -silmukat, 277–80
For Output -avainsana, 307
ForeColor-ominaisuus, 25
ForLoop-ohjelma, 164
Form Layout -ikkuna, 47
Form_Load-aliohjelma, 38, 58–59
Forms-kokoelma, 282
.frm-tiedostot, 34
funktiot
- argumentit, 115
- järjestelmäkello, 88
- kirjoittaminen, 258–64
- kutsuminen, 258–59, 260
- kuvaus, 115
- luettelot, 88, 127
- matemaattiset, 127
- muoto, 259
- MyRate-ohjelmassa, 260–264
- yleiskäyttöisinä aliohjelmina, 258

G

graafiset komentopainikkeet, 219–26
grafiikka lomakkeissa, 214–18
grafiikkametodit, 218
grafiikkatiedostot
- määrittäminen kuvakehyksessä, 26–27
- näyttömuodot, 48

Grid-objekti, 72, 73
Grid32.ocx-tiedosto, 72
GrowFont-ohjelma, 166

H

haarautuminen. ***Katso myös*** **If…Then-ehtorakenteet**
- Select Case -rakenteissa, 145, 147
- vianmääritys, 156

hakemistoluetteloruudut, 48, 49, 50
harjoitustiedostot
- asentaminen, xxiii
- käyttö, xix–xxii
- ohjeita, xxiii
- Open Project -valintaikkunassa, 6

Height-ominaisuus, 239–40
Hello-ohjelma, 46
Hide-metodi, 190, 195
hiiren osoitin
- kuvaus, 233
- muuttaminen, 233
- painikkeiden nimien näyttäminen, 9

Hour-funktio, 88

I

If…Then -ehtorakenteet, 138–45
IfBug-ohjelma, 152–57

ikkunat. ***Katso myös*** **Code-ikkuna; lomakkeet**
koon muuttaminen, 8
Project-ikkuna, 12–13
Properties-ikkuna, 9–11, 22, 23
siirtäminen, 9
telakoiminen, 8, 11
Image-objekti, 21, 50, 91–92, 167, 219, 313
ImageList-objekti, 74
Index-ominaisuus, 170, 171
.inf-tiedostot, 302
.ini-tiedostot, 302
InputBox-funktio, 114–15, 319
InputBox-ohjelma, 113–15
Insert Object -valintaikkuna, 62–63, *62,* **336**
Integer-tietotyyppi, 118
Interval-ominaisuus, 177, 178, 179
Italian-ohjelma, 191–93. ***Katso myös*** **MyItalian2-ohjelma**
Italian2-ohjelma, 191, 196

J

jakojäännösoperaattori (Mod), *123,* **127–29,** *130*
jakolaskuoperaattori (/), 123, 124–26, *130*
julkiset funktiot. ***Katso*** **funktiot**
julkiset muuttujat, 121, 253–57
järjestelmäkellon funktiot, 88

K

kaatuminen, ohjelman, 53
kaaviot. ***Katso*** **Excel-kaaviot OLE-objekteina**
kaksoistarkkuuden liukulukutyyppi, 118
Kello-valikko
komentojen järjestyksen muuttaminen, 82–83
komentojen lisääminen, 79–80
luominen, 78–80
MyMenu-ohjelman suorittaminen, 86–87
MyMenu-ohjelman tallentaminen, 86
pikanäppäinten määrittäminen, 101–2
vaihtoehtojen käsitteleminen, 83–86
valintakirjainten lisääminen, 80–82
kentät, 67
kertaustehtävät, xxii
kertolaskuoperaattori (*), *123,* 124–26, *130*
Keskeytä-painike, 153
kielioppivirheet, 151
Kirjahaku-valintaikkuna, 317–18
kokoelmat
merkintätapa, 276
objekteihin viittaaminen, 276
siirtäminen, 277–79
yleiskuvaus, 275
kokonaislukujen jakolaskuoperaattori (\~), 123, 127–30, *130*
kokonaislukutyyppi, 118
komentojen poistaminen käytöstä, 93
komentopainikkeet
eritilaiset, 219–26
graafiset, 219–26
koon muuttaminen, 19
kuvaus, 46, 357
lisääminen lomakkeisiin, 18, 19, 45–46
luominen, 18, 19, 219–26
ohjelmalauseiden kirjoittaminen, 29–32
ominaisuuksien muuttaminen, 10–11, 22–23, 46
poistaminen, 19
siirtäminen, 19
kommentit, 33
koon muuttaminen
ikkunat, 8
komentopainikkeet, 19
lomakkeet, 17
objektit, 19
koonmuutto-osoitin, 334

koordinaatistot
objektien siirtäminen, 235
yleiskuvaus, 234–35
kuvakehykset
graafiset komentopainikkeet, 219, 221
grafiikkamuodot, 48
kuvaus, 21
lisääminen lomakkeisiin, 21–22, 50, 91–92, 167, 221, 239
objektitaulukossa, 167, 168
ominaisuuksien muuttaminen, 26–27
tyhjentäminen, 97
valitseminen, 168
kuvakehysobjekti, 236, 313
kuvaketiedostot, 48, 167, 168–69
kuvat. *Katso myös* kuvakehykset
lataaminen suorituksen aikana, 51, 97, 168
lisääminen lomakkeisiin, 21–22
luominen Viiva- ja Kuvio-työkaluilla, 214–18
kuvien lataaminen suorituksen aikana, 51, 97, 168
kuviot lomakkeissa, 214–18
Käynnistä-painike (Visual Basicin työkalurivi), 7, 34, 52
Käynnistä-painike (Windows 95), 4
käynnistäminen
ohjelma, 34
uusi projekti, 17
Visual Basic, 4–5
käyttäjän määrittämät tietotyypit, 121
käyttöliittymä
lomakkeet, 8–9
suunnitteluohjeita, 357–58

L

Label-objekti, 19, 68, 83, 313
lainausmerkit (“), 111
laskentajärjestys, operaattoreiden, 130–31
laskurimuuttuja
For…Next-silmukoissa, 163, 166, 169
näyttäminen Print-metodilla, 162
virheilmoituksissa, 209
lataaminen, projektit, 37–38
lausealiohjelmat
kirjoittaminen, 264–70
kutsuminen, 265–66
kuvaus, 258
luominen, 266–69
muoto, 264
lauseen syntaksi, 108
lausekkeet
funktiot, 127
kehittyneet operaattorit, 127–30
kuvaus, 123
operaattoriluettelo, 123
perusoperaattorit, 124–26
sulkeet, 131
lausekkeet. *Katso* ehtolausekkeet
Left-ominaisuus, 234, 235, 238
levykeasemat, virheet, 205–6
Liitä määräten -valintaikkuna, 338–39
Line Input -avainsana, 303, 306
Line-metodi, 218
Line-objekti
StartFrm-ohjelmassa, 215–17
yleiskuvaus, 214
linkitetyt objektit, 336
ListBox-objekti, 147, 313
ListIndex-ominaisuus, 58, 149
ListView-objekti, 74
Lisää päivämäärä -komento, 308, 310
Load Picture -valintaikkuna, 26–27, *26*
Load-lause, 189, 190
LoadPicture-funktio, 51, 97, 168
Login-ohjelma, 141
.log-tiedostot, 302
lomakemoduulit, 250
lomakkeet
esimerkki, 5
hierarkia, 197

komentopainikkeiden lisääminen, 18, 19, 45–46
koon muuttaminen, 17
koordinaatistot, 234–35
kuvaus, 8–9
kuvien lisääminen, 21–22
kuvioiden lisääminen, 214–18
käsittely tapahtuma-aliohjelmissa, 194–97
käyttötarkoituksia, 187–88
lataaminen ja poistaminen, 189–90
lisääminen ohjelmiin, 187–88, 191
MDI (Multiple Document Interface), 197
näyttäminen, 189–90
objekteina, 187
objektien lisääminen, 18–22
objektien poistaminen, 19
oletuslomake, 8
otsikoiden lisääminen, 19–21, 83–84, 221
piilottaminen Hide-metodilla, 190, 195
piilottaminen Visible-ominaisuudella, 174
rivien lisääminen, 214–18
ruudukko, 19
tulostaminen, 198–203
tyhjät, 17
useat, 187–88
uudelleenkäyttö, 191
vaihtaminen, 195
Long Integer -tyyppi, 118
loogiset operaattorit
ehtolausekkeissa, 143–45
kuvaus, 143
luettelo, 143
loogiset virheet, 151
Lopeta-painike, 47
lopettaminen, Visual Basic, 37
lopetusehto, 174
loputtomat silmukat, 174
luetteloruudut
kuvaus, 56
ohjelmalauseiden katseleminen, 58
syöttötietojen käsittely, 147–50
tuetut tapahtumat, 137
vaihtoehtojen lisääminen, 59, 60, 148–49
luokkamoduulit, 250

M

Make Project -valintaikkuna, 36, *36*
.mak-tiedostot, 6
matemaattiset funktiot. ***Katso*** **funktiot**
matemaattiset operaattorit. ***Katso*** **operaattorit**
MDI-lomakkeet, 197, 220
Menu Editor
esimerkkejä, *78, 80, 93*
komentojen järjestyksen muuttaminen, 82–83
komentojen lisääminen valikoihin, 79–80, 83
komentojen poistaminen, 83
kuvaus, 77
käynnistäminen, 78
lopettaminen, 80
pikanäppäinten määrittäminen, 101–2
valikoiden lisääminen, 77–83, 92–93
valikoiden luominen, 78–80
valintaikkuna, 78–80, *78*
valintakirjainten määrittäminen, 80–82
Menu-ohjelma, 87
merkkijonojen yhteenliittämisoperaattori (&), *123,* **127–30, 311**
merkkijonot, 111, 114, 123, 126, 129
merkkijonotyyppi, 118
metatiedostot, 26
metodi, kuvaus, 59
Microsoft Access -tietokannat, 66–70, 312
Microsoft Excel. ***Katso*** **Excel-kaaviot OLE-objekteina; Excel-taulukot**
Microsoft FoxPro -tietokannat, 312
Microsoft Jet Data Access Engine, 312
Microsoft SQL Server, 312
Microsoft Word

asiakirjaobjektin lisäys ohjelmaan, 62–63
Find-objekti, 351–52
käynnistäminen MyOleBid-ohjelmasta, 65
objektikirjasto, 343–46
versiot, 343, 347
Minute-funktio, 88
***mnu*-etuliite, 79**
Mod-operaattori, *123,* 127–30, *130*
moduulit, 121, 249–50. *Katso myös* vakiomoduulit
moniulotteiset taulukot, 293–97
Month-funktio, 88
moodiset lomakkeet, 189
moodittomat lomakkeet, 189
MouseDown-aliohjelma, 219, 221, 223–25
MouseIcon-ominaisuus, 233
MouseMove-aliohjelma, 219
MousePointer-ominaisuus, 233
MouseUp-aliohjelma, 219
Move-metodi, 235, 238
Move-ohjelma, 277
MsgBox-funktio, 116–17, 320
muistin vapauttaminen, 349
mukautetut objektit. *Katso* ActiveX-objektit
Multiple Document Interface (MDI) -lomakkeet, 197, 220
MusicEIS-ohjelma, 340
muunnelmat, 109
muunnelmatyyppi, 118
muuttujat
arvoargumentit, 271–72
arvojen muuttaminen, 110–12
esittelyt, 109, 110, 111
julkiset, 121, 253–57
kuvaus, 51, 59
käyttö ohjelmissa, 110–17
nimeämissääntöjä, 113
paikalliset ja julkiset, 249–50
taulukoissa, 282–93
tietojen tallentaminen, 113–15
tietotyypit, 109, 118
tulostaminen, 116–17
viittausargumentit, 264
yleiskuvaus, 108–9
MyAddData-ohjelma
luominen, 319–21
muokkaaminen MyDelData-ohjelmaksi, 322–24
suorittaminen, 321–22
MyBackup-ohjelma
luominen, 325–26
suorittaminen, 326
MyButtons-ohjelma
luominen, 221–25
suorittaminen, 225–26
MyCase-ohjelma
luominen, 147–49
suorittaminen, 149
MyCelsius-ohjelma
luominen, 174–75
suorittaminen, 175–76
MyCtlArray-ohjelma
luominen, 167–69
muokkaaminen MyStepLoop-ohjelmaksi, 170–72
suorittaminen, 169–70
tallentaminen MyStepLoop-ohjelmaksi, 172
MyData-ohjelma
luominen, 67–70
suorittaminen, 70
MyDef.frm-tiedosto, 194
MyDelData-ohjelma
luominen, 322–23
suorittaminen, 324
MyDialog-ohjelma
kuvakehysobjektien lisääminen, 91–92
luominen, 89–98
suorittaminen, 98–100
valikkokomentojen lisääminen, 92–93
valikkokomentojen poistaminen käytöstä, 93
valikoiden lisääminen, 92–93

yleisten valintaikkunaobjektien lisääminen, 91
yleisten valintaikkunaobjektien muokkaaminen, 94–98
MyDigClock-ohjelma
luominen, 177–78
suorittaminen, 178–79
MyDragDrop-ohjelma
animaation lisääminen, 236–39
luominen, 228–31
suorittaminen, 231–33
MyDynArray-ohjelma
luominen, 291–93
suorittaminen, 293
MyFinalErr-ohjelma
luominen, 205–6
suorittaminen, 207–8
MyFindData-ohjelma
luominen, 316–17
suorittaminen, 317–19
MyFixArray-ohjelma
luominen, 287–89
suorittaminen, 289–90
MyForLoop-ohjelma
luominen, 163–64
muokkaaminen MyGrowFont-ohjelmaksi, 164–66
suorittaminen, 164
MyGrowFont-ohjelma
luominen, 164–65
suorittaminen, 165–66
MyHello-ohjelma
luominen, 44–47
suorittaminen, 47–48
taustaa, 44
MyInputBox-ohjelma
luominen, 113–14
suorittaminen, 114–15
MyItalian2-ohjelma
lomakkeen luominen, 192–95
lomakkeen näyttäminen, 195–96
suorittaminen, 195–97
tulostimen tuen lisääminen, 198–200
MyLogin-ohjelma
luominen, 140–41
muokkaaminen MyPass-ohjelmaksi, 144
suorittaminen, 142–43
MyLucky-ohjelma
käyttöliittymän luominen, 17–22
lataaminen, 37–38
lisäysten tekeminen, 37–38
muodostaminen, 15–33
ohjelmalauseiden kirjoittaminen, 29–32
ohjelmoinnin vaiheet, 15–16
ominaisuuksien määrittäminen, 22–28
osat, 15–16
pikakuvakkeen luominen, 37
suoritettavan tiedoston luominen, 35–37
suorittaminen, 34–35, 37
tallentaminen levylle, 33–34
MyMenu-ohjelma
luominen, 78–86
suorittaminen, 86–87
tallentaminen MyDialog-ohjelmaksi, 89
MyMove-ohjelma
luominen, 277–78
muokkaaminen MyTag-ohjelmaksi, 279–81
suorittaminen, 278–79
MyMusicEIS-ohjelma
luominen, 334–35
suorittaminen, 340–41
MyNameConv-ohjelma
luominen, 243–44
suorittaminen, 244–45
MyOleBid-ohjelma
luominen, 61–65
suorittaminen, 65–66
MyPass-ohjelma
luominen, 144
suorittaminen, 144–45

MyPrintFrm-ohjelma
luominen, 198–99
suorittaminen, 200–201
MyRate-ohjelma
luominen, 260–63
suorittaminen, 263–64
MySmoke-ohjelma
luominen, 236–38
suorittaminen, 238
MyStartFrm-ohjelma
luominen, 215–17
suorittaminen, 217–18
MyStepLoop-ohjelma
luominen, 170–71
suorittaminen, 171–72
MyTag-ohjelma
luominen, 279–80
suorittaminen, 281
MyTeams-ohjelma
luominen, 264–69
suorittaminen, 270
MyTestMod.bas-tiedosto, 251–52
MyTimePass-ohjelma
luominen, 179–81
suorittaminen, 181
MyUseWord-ohjelma
luominen, 347–49
suorittaminen, 349–50
MyVarTest-ohjelma
luominen, 110–12
suorittaminen, 112
MyWins-ohjelma
luominen, 254–56
MyWordList.frm-tiedosto, 194
suorittaminen, 256–57
MyZoom-ohjelma
luominen, 239–40
suorittaminen, 240

N

Name-ominaisuus, 241, 243, 252
NameConv-ohjelma, 244
New Project -komento, 17
New Project -valintaikkuna, *4*
nimeämissääntöjä
muuttujat, 113
objektit, 242–43
Not-operaattori, 143, 144
Now-funktio, 88
näppäimet
pikanäppäimet, 101–2
valintakirjaimet, 80–82
Näytä objekti -painike, 54

O

Object Browser -ohjelma
käynnistäminen, 344
valintaikkuna, 344–45, *344, 345*
Word-objektien katseleminen, 343–46
yleiskuvaus, 343
objektien nimeäminen, 241–45, 252
objektien nimien vaihtaminen, 195
objektien valitseminen
lomakkeesta, 22, 23
objektitaulukosta, 168
Properties-ikkunasta, 23
ryhmänä, 168
objektikirjastot
katseleminen Object Browserilla, 343–46
viittausten lisääminen, 343–44, 346, 347
Objects-kokoelma, 275, 276, 277–80
objektisovellukset, 342
objektit. *Katso myös* ActiveX-objektit; painikkeet
kuvaus, 58
mukautetut ja ActiveX, 71
nimien katseleminen, 9
vakio-, 313
työkaluruudussa, 9
yleiskuvaus, 9, 43, 44

objektit. *Katso myös* kokoelmat; dataobjektit; OLE-objektit
koon muuttaminen, 19
kuvaus, 58
linkitetyt ja upotetut, 336
lisättävät, 73
lisääminen lomakkeisiin, 18–22
nimen vaihtaminen, 195
nimeäminen, 241–45
nimeämissääntöjä, 242
näkyvät ja näkymättömät, 27, 174, 203, 214, 228, 231
ohjelmalauseiden kirjoittaminen, 29–32
ohjelmoiminen Automation-tekniikalla, 341–50
ominaisuuksien muuttaminen, 10–11
poistaminen lomakkeista, 19
Properties-ikkunassa, 9–10
siirtäminen lomakkeessa, 19
siirtäminen Move-metodilla, 235
taulukoina, 167
tiedostojärjestelmä-, 48–53
toiminnot, 28
tuetut tapahtumat, 137
valitseminen lomakkeesta, 22, 23
valitseminen Properties-ikkunasta, 23
valitseminen ryhmänä, 168
objektitaulukot. *Katso myös* taulukot
kuvaus, 167
luominen, 167
objektien valitseminen ryhmänä, 168
.ocx-tiedostot, 71, 72
ODBC-tietokannat, 312
ohjaavat sovellukset, 342
ohje
virhetilanteissa, 151
Word-objektikirjasto, 345–46
yleiskuvaus, 13–15
ohjelmalauseet
kirjoittaminen, 358
muoto, 108, 359
muuttujat, 110–12
yleiskuvaus, 28–30, 108
ohjelman kirjoittaminen, 358
ohjelmat
algoritmien luominen, 16, 356
jakeleminen, 360
kaatuminen, 53
kuvaus, 355
käyttöliittymän suunnitteleminen, 17–22, 357–58
kääntäminen, 35–37, 360
lataaminen, 37–38
lisäysten tekeminen, 37–38
muodostaminen, 15–33, 358
ohjeiden etsiminen, 13–15
suorittaminen, 7, 34–35, 37
suunnitteleminen, 15–16, 355–56
tallentaminen, 33–34
testaaminen, 151–57, 360
tulostintuen lisääminen, 196–200
vaiheet, 15–16, 358
ohjelmien kääntäminen, 35–37, 360
ohjelmien suorittaminen, 7, 34–35, 37
ohjelmointisääntöjä, 30
ohjelmointityyli, 30
oikeinkirjoituksen tarkistus. *Katso* CheckSpelling-metodi
OLE Automation. *Katso* Automation-tekniikka
OLE-objektit. *Katso myös* sovellusobjektit
lisääminen ohjelmiin, 62–64, 333, 336
sovellusten käynnistäminen, 61–65
OleBid-ohjelma, 65
oletettu esittely, 109–10
oletuslomake, 8
ominaisuudet
asetukset, 9–10
kuvaus, 58–59
luettelo, 27–28
muuttaminen Properties-ikkunassa, 10–11
muuttaminen silmukassa, 164–66
muuttaminen suorituksen aikana, 46

muuttaminen suunnittelun aikana, 10–11, 22–28
yleiskuvaus, 11
ominaisuuksien määrittäminen. ***Katso*** **ominaisuudet**
ominaisuusaliohjelmat, 258
On Error -lause
DriveErr-ohjelmassa, 205
yleiskuvaus, 204
Onnen seiska -ohjelma, 34, 253, 260
Open Database Connectivity (ODBC) -tietokannat, 312
Open Project -valintaikkuna, 6, *6*
Open-lause, 302–3, 307
Open-valintaikkuna
kuvaus, 89, 357
lisääminen, 93–94
MyDialog-ohjelmassa, 98
tapahtuma-aliohjelma, 94–96
Tekstiselain-ohjelmassa, 304–5
operaattorit
kuvaus, 123
laskentajärjestys, 130–31
loogiset, 143–45
vertailu-, 138–40, 146–47
Option Base -lause, 287, 288
Option Explicit -lause, 110
Or-operaattori, 143, 144
Ostospaikka-ohjelma
kuvaus, 54
lauseiden katseleminen, 57–59
suorittaminen, 54–56
otsikot
lisääminen lomakkeisiin, 19–21, 83–84, 221
ominaisuuksien muuttaminen, 23–25

P

painikkeet. ***Katso myös*** **objektit**
graafiset, 219–26
koon muuttaminen, 19
kuvaus, 46, 357
lisääminen lomakkeisiin, 18, 19, 45–46
luominen, 18, 19, 219–26
nimien katseleminen, 9
ohjelmalauseiden lisääminen, 29–32
ominaisuuksien muuttaminen, 10–11, 22–23, 46
poistaminen, 19
siirtäminen, 19
Paint-ohjelma
käynnistäminen MyOleBid-ohjelmasta, 66
objektien lisääminen Projektisuunnittelu-ohjelmaan, 64
palvelinsovellukset, 342
Paradox-tietokannat, 312
Password-ohjelma, 144
Path-ominaisuus, 50, 51
Pattern-ominaisuus, 50–51
Perusoperaattorit-ohjelma, 124–26
perustietotyypit, 118–21
peräkkäistiedostot, 303
Picture-ominaisuus, 26, 222, 225
pienempi kuin -operaattori (<), 138
pienempi tai yhtä suuri kuin -operaattori (<=), 138
piilottaminen, lomakkeet, 174, 190, 195
piirrokset, lisääminen Viiva- ja Kuvio-työkaluilla, 214–18
pikakuvakkeet, 37
pikanäppäimet, 101–2
Pikaseuranta-painike, 155
pilkku (,), 164
pitkä kokonaislukutyyppi, 118
Poista tietue -valintaikkuna, *324*
poistaminen
komentopainikkeet, 19
objektien tekstit, 25
objektit, 19
tietueet, 322–24
valikkokomennot, 83
Print # -lause, 308, 311

Print-metodi, 162–64, 196, 288
Print-valintaikkuna, 89
Printer-objekti
lisääminen ohjelmiin, 196–200
metodiluettelo, *198*
ominaisuusluettelo, *198*
yleiskuvaus, 196–98
Printers-kokoelma, 282
PrintForm-metodi, 201–3
PrintFrm-ohjelma, 200
Private Sub -lause, 29, 30
Prntout2.wmf-tiedosto, 206, 207
ProgressBar-objekti, 74
Project Properties -valintaikkuna, 36
Project-ikkuna, 12–13, *13*
projektitiedostot
avaaminen, 6–7
kuvaus, 12
tallentaminen, 34
Properties-ikkuna, 9–11, *10*, 22, *22*, 23, *23*
PSet-metodi, 218
puolipiste (;), 164

Q

QNote-ohjelma
lauseiden katseleminen, 311
suorittaminen, 308–10

R

Randomize-lause, 38
Rate-ohjelma, 263
ReadOnly-ominaisuus, 313, 319, 322
Recordset-objekti, 316–19
RecordSource-ominaisuus, 69, 315
References-valintaikkuna, 343–44
Resume-lause, 204–5
RichTextBox-objekti, 74
rivinjatkomerkki (_), 141
rivit, lisääminen lomakkeisiin, 214–18
Rnd-funktio, 127
Run-valintaikkuna, xviii, *xviii*
ruudukot, 19

S

salasanat
aikarajan asettaminen, 179–81
lisääminen Login-ohjelmaan, 144–45
Save Project As -valintaikkuna, *34*
SearchStr$-funktio, 317
Second-funktio, 88
Seek-metodi, 317
Select Case -ehtorakenteet
ehtolausekkeet, 146–47
haarautuminen, 145, 147
luetteloruudun käsittely, 147–50
muoto, 145
vertailuoperaattorit, 146–47
Set-lause, 349
SetFocus-metodi, 320
Sgn-funktio, 127
Shape-objekti, 214, 215–17
Shape-ominaisuus, 214
Show-metodi, 189, 190
ShowColor-metodi, 89
ShowFont-metodi, 89
ShowOpen-metodi, 89, 94, 95, 302
ShowPrinter-metodi, 89
ShowSave-metodi, 89, 307
ShowText-ohjelma
lauseiden katseleminen, 306–7
suorittaminen, 303–5
sidotut objektit
kuvaus, 312
tietokannan tietojen näyttäminen, 312–13
siirtäminen
kokoelmat, 277–79
komentopainikkeet, 19
lomakkeen objektit, 19
objektien siirtäminen Move-metodilla, 235, 238
työkaluruutu, 9

sijoitusoperaattori (=), 109
silmukat
Do-, 173–76
For…Next-, 161–72
loputtomat, 174
ominaisuuksien muuttaminen, 164–66
virheet, 208
Sin-funktio, 127
sisentäminen, lauseet, 30
Slider-objekti, 74
Smoke-ohjelma, 238
Source-muuttuja, 231
sovellukset
asiakirjakeskeiset, 197
käynnistäminen OLE-objekteilla, 61–65
sovellusobjektit. *Katso myös* OLE-objektit
esimerkkejä, 333
katseleminen Object Browser -ohjelmalla, 343–46
kokoelmissa, 282
lisääminen ohjelmiin, 336–40
ohjelmoiminen Automation-tekniikalla, 341–50
päivittäminen, 341
yleiskuvaus, 332–33
sovellusten käynnistäminen OLE-objekteilla, 61–65
SQL Server, 312
Sqr-funktio, 127
StartFrm-ohjelma, 218
StatusBar-objekti, 74
Step-avainsana, 170–72
StepLoop-ohjelma, 172
StepUp-ohjelma
animaatio, 239
kuvaus, 7
käyttöliittymä, 7
lomake, 6–7
Move-metodi, 239
ominaisuusasetusten muuttaminen, 10–11
projektitiedostojen avaaminen, 6–7
suorittaminen, 7
tiedostoluettelon katseleminen, 12–13
Stop-lause, 157
Str-funktio, 127
String-tietotyyppi, 118
Sulje-painike, 47
sulkeet, 131
suoritettavat tiedostot, 12, 35–37, 360
suurempi kuin -operaattori (>), 138
suurempi tai yhtä suuri kuin -operaattori (>=), 138
syntaksi, 30, 108, 359
sääntöjä
muuttujien nimeäminen, 113
objektien nimeäminen, 242–43
ohjelmoiminen, 30

T

TabStop-ominaisuus, 193
Tabstrip-objekti, 74
Tag-ohjelma, 281
Tag-ominaisuus, 222, 224, 229, 231, 279–80
tallentaminen
ohjelmat, 33–34
projektitiedostot, 34
Tan-funktio, 127
tapahtuma-aliohjelmat. *Katso myös* yleiskäyttöiset aliohjelmat
ja tapahtumaohjattu ohjelmointi, 136–37
kuvaus, 29, 46, 51, 59
lomakkeiden käsittely, 194–97
muuttujat, 114, 116–17
tietotyypit, 120–21
vakiot, 122–23
valikko-objektit, 83–86
yleiset valintaikkunat, 94–98
tapahtumaohjattu ohjelmointi, 135–36
tapahtumat, 135–37
taulukot. *Katso myös* objektitaulukot
dynaamiset, 284, 290–93

esitteleminen, 283–85
käyttö, 285–86
luominen, 283–85
moniulotteiset, 293–97
muoto, 284
MyFixArray-ohjelma, 287–90
vakiokokoiset, 284–85
yleiskuvaus, 282–83
Teams-ohjelma, 270
tehtäväpalkki, Windows 95, 5
tekstiruudut
kuvaus, 45, 54
lausealiohjelman luominen, 266–69
lisääminen lomakkeisiin, 44–45, 68
luominen, 44–45, 68
ominaisuuksien muuttaminen, 46
salasanan kysyminen, 145
tekstitiedostojen näyttäminen, 302–7
tietokannan sisällön näyttäminen, 313–16
tekstitiedostot
avaaminen syöttöä varten, 302–7
kuvaus, 302
luominen, 307–11
näyttäminen, 302–7
telakoiminen, 8, 11
testaaminen, 151–57, 360
Text-ominaisuus, muuttaminen suorituksen aikana, 46
TextBox-objekti, 44, 45, 68, 313
tiedostojärjestelmäobjektit, 48–53
tiedostoluetteloruudut, 48, 49–50
tiedostot. *Katso* projektitiedostot; tekstitiedostot
tietokannat
kuvaus, 66
mukautettujen sovellusten luominen, 312
tietojen etsiminen, 316–19
tietueiden lisääminen, 319–22
tietueiden muokkaaminen, 71
tietueiden näyttäminen, 68, 70
tietueiden poistaminen, 322–24
varmistuskopioiminen, 325–26
tietotyypit
käyttäjän määrittämät, 121
käyttö ohjelmassa, 119–21
luettelo, 118
luominen, 121
määrittäminen, 119
perus-, 118–21
taulukoiden, 283
tietueet
katseleminen tietokannassa, 68, 70
kuvaus, 67
lisääminen tietokantaan, 321–22
muokkaaminen tietokannassa, 71
poistaminen tietokannasta, 324
Time-funktio, 85, 88
TimePass-ohjelma, 181
Timer-objekti, 177, 236
Toolbar-objekti, 74
Top-ominaisuus, 235, 236, 238
totuusarvoiset lausekkeet, 138
TreeView-objekti, 74
TrueType-fontit, 198
tulostaminen
lomakkeiden tulostaminen PrintForm-metodilla, 201–3
rivit Print-metodilla, 196–98
tulostimen tuen lisääminen ohjelmiin, 196–200
.txt-tiedostot, 302
twip-yksikkö, 234
tyhjät lomakkeet, 188
Type-lause, 121
työkalurivit
painikkeiden toimintojen näyttäminen, 5
siirtäminen, 9
telakoiminen, 8, 11
työkaluruutu
ActiveX-objektien lisääminen, 72–73
esimerkki, *5, 73*
kuvaus, 9
siirtäminen, 9
telakoiminen, 8, 11

U

Unload-lause, 190
Until-avainsana, 176
Update-metodi, 341
upotetut objektit, 336
UseWord-ohjelma, 350

V

vakio-objektit, 313
vakiokokoiset taulukot, 284–85, 287–90
vakiomoduulit
julkiset muuttujat, 253–57
lisääminen projekteihin, 250–52
luominen, 250–52
poistaminen projekteista, 252
yleiskuvaus, 249–50
yleiskäyttöiset aliohjelmat, 257–58
vakiot, 122 23
Val-funktio, 125–26, 127, 130
valikkokomennot
järjestyksen muuttaminen, 82–83
lisääminen valikoihin, 79–80, 83
ottaminen käyttöön ja poistaminen käytöstä, 93
pikanäppäinten määrittäminen, 101–2
poistaminen, 83
tapahtuma-aliohjelmat, 83–86
valintakirjainten määrittäminen, 80–82
valikkorivi
esimerkki, 5
kuvaus, 4
käyttö, 6
valikot
komentojen järjestyksen muuttaminen, 82–83
komentojen lisääminen, 79–80, 92–93
kuvaus, 77, 357
lisääminen lomakkeisiin, 77–80, 92–93
luominen, 78–80
Menu Editorin käyttö, 78–83, 92–94
objekteina, 83
ohjelmalauseiden kirjoittaminen, 84–86
pikanäppäinten määrittäminen, 101–2
sääntöjä, 81
tapahtuma-aliohjelmat, 83–86
valintakirjainten määrittäminen, 80–82
valintaikkunat. *Katso myös* yleiset valintaikkunat; InputBox-funktio; MsgBox-funktio
objekteina, 54
omien lisääminen, 101
valintakirjaimet, lisääminen valikkokomentoihin, 80–82
valintanapit, 55
valintaruudut, 56, 57
Value-ominaisuus, 57
Variant-tyyppi, 118
VarTest-ohjelma, 110
.vbg-tiedostot, 6
.vbp-tiedostot, 6, 12
.vbx-tiedostot, 71
vedon ja pudotuksen tuki
käyttötarkoituksia, 233
lisääminen ohjelmaan, 227–33
yleiskuvaus, 227–28
vertailuoperaattorit
ehtolausekkeissa, 138, 139–40, 146–47
esimerkkejä ehtolausekkeista, 138
If…Then-rakenteissa, 138–40
luettelo, 138
Select Case -rakenteissa, 145–50
vianmääritys, 151–57
vierityspalkit, 357
viivat, lisääminen lomakkeeseen, 214–18
virheenkäsittelijät
ja virheilmoitukset, 205–8
käyttötarkoituksia, 203–4
luominen, 205–6
On Error -lause, 204
uudelleenyritysten rajoittaminen, 209
yleiskuvaus, 53, 203

virheet
ajonaikaiset, 53, 203–9
etsiminen, 151–57
ja virheenkäsittelijät, 203–9
tyypit, 151
virheiden korjaaminen, 151–57
virheilmoitukset, 205–8
virhesilmukat, 208
Visible-ominaisuus, 27, 174, 203, 214, 228, 231
Visual Basic
käynnistäminen, 4–5
lopettaminen, 37
Object Browser -ohjelma, 343–46
ohjejärjestelmä, 13–15
ohjelmien suorittaminen, 7
ohjelmointikieli, 359
ohjelmointiympäristö, 3–15
ohjelmointityökalut, 4–7
vähennyslaskuoperaattori (–), *123,* 124–26, *130*
Väri-valintaikkuna
esimerkki, *99*
kuvaus, 89
lisääminen, 93, 94, 96–97

W

Watches-ikkuna, 151, *155,* 156, 157, *158*
WdSearch-ohjelma, 351–52
Weekday-funktio, 88
While-avainsana, 173, 176
Width-ominaisuus, 239–40
Windows 95
ActiveX-objektit, 74
Käynnistä-painike, 4
tehtäväpalkki, 5
WindowState-ominaisuus, 190
.wmf-tiedostot, 48, 98
Word for Windows. *Katso* Microsoft Word
Word Visual Basic, 346

X

Xor-operaattori, 143, 144

Y

Year-funktio, 88
yhteenlaskuoperaattori (+), *123,* 124–26, *130*
yhteenliittämisoperaattori (&), *123,* 127–30, 311
yhtäläisyysoperaattori (=), 138
yksinkertaisen tarkkuuden liukulukutyyppi, 118
yleinen ActiveX-valintaikkunaobjekti, 72, 73
yleiset valintaikkunat
ja Open-lause, 302–3
kuvaus, 88
lisääminen lomakkeisiin, 91
metodit määrittämistä varten, 88–89
tapahtuma-aliohjelmat, 94–98
yleiskäyttöiset aliohjelmat
funktioaliohjelmien kirjoittaminen, 258–64
lausealiohjelmien kirjoittaminen, 264–70
tyypit, 258
yleiskuvaus, 257–58
ylilomakkeet, 197
yrityksen tietojärjestelmä
käyttötarkoituksia, 332
luominen, 331–41
MusicEIS-sovelluksen luominen, 334–40
OLE-objektin käyttö, 333–41
suorittaminen, 340–41
yleiskuvaus, 331–32

Z

Zoom-ohjelma, 240

Tämän kirjan käsikirjoitus valmisteltiin ja lähetettiin Microsoft Pressille elektronisessa muodossa. Tekstitiedostot tehtiin Microsoft Word 6.0 for Windowsilla. Sivut taittoi Microsoft Press Adobe PageMaker 6.01 for Windowsilla käyttäen leipätekstissä Stone Informal-fonttia ja otsikoissa Lucida Sans Bold -fonttia. Taitetut sivut lähetettiin kirjapainoon painovalmiina tiedostoina.

Kannen suunnittelija
Microsoft Press Creative Department

Valokuvaaja
Robin Bartholick

Sisäpuolen graafinen suunnittelija
Pam Hidaka

Sisäpuolen graafinen taiteilija
Michael Victor

Päätaittaja
Peggy Herman

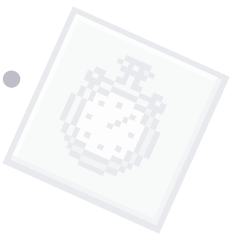
Hakemiston tekijä
Julie Kawabata

Blank page 384

Step by Step

Harjoituslevy

Oheinen CD-tietolevy sisältää käyttövalmiita harjoitustiedostoja, jotka täydentävät tämän kirjan harjoituksia. Siinä on myös neljä lisäharjoitusta, jotka voit tehdä milloin vain. Voit avata nämä tiedostot, jos sinulla on Microsoft Visual Basic 5.0 Learning Edition, Professional Edition tai Enterprise Edition. Tarvitset myös joko Microsoft Windows 95 -käyttöjärjestelmän tai Windows NT -käyttöjärjestelmän version 3.51 Service Pack 5:n tai uudemman. Harjoituksen 12 tehtävissä tarvitset myös Microsoft Office 97 Standard tai Professional Editionin.

Useimmissa tämän kirjan harjoituksissa käsitellään harjoituslevyn tiedostoja. Ennen kuin aloitat harjoitukset, lue kirjan alussa oleva kohta Harjoitustiedostojen asentaminen ja käyttö. Siinä on jokaisen tiedoston kuvaus ja ohjeet, joiden mukaan voit asentaa tiedostot tietokoneesi kiintolevylle.

Lue edellisellä sivulla oleva lisenssisopimus ennen kuin avaat CD-tietolevyn suojataskun.